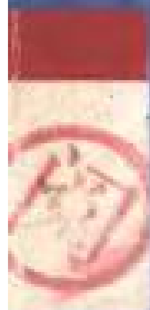


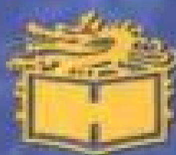
面向21世纪课程教材

Textbook Series for 21st Century

数学实验

李尚志 陈发来 吴耀华 张韵华 著





高等教育出版社

HIGHER EDUCATION PRESS



面 向 2 1 世 纪 课 程 教 材
Textbook Series for 21st Century

数学实验

李尚志　陈发来　吴耀华　张韵华　著





高 等 教 育 出 版 社
HIGHER EDUCATION PRESS

（京）112 号

图书在版编目(CIP)数据

数学实验/李尚志等著. －北京 ：高等教育出版社，
1999
ISBN 7-04-007762-0

Ⅰ. 数… Ⅱ. 李… Ⅲ. 数学-实验 Ⅳ. 01-33

中国版本图书馆 CIP 数据核字(1999)第 37266 号

数学实验
李尚志 陈发来 吴耀华 张韵华 著

出版发行	高等教育出版社		
社　　址	北京市东城区沙滩后街 55 号	邮政编码	100009
电　　话	010—64054588	传　　真	010—64014048
网　　址	http://www.hep.edu.cn		

经　　销	新华书店北京发行所
排　　版	高等教育出版社照排中心
印　　刷	国防工业出版社印刷厂
纸张供应	山东高唐纸业集团总公司

开　　本	787×960　1/16	版　　次	1999 年 9 月第 1 版
印　　张	13.75	印　　次	1999 年 9 月第 1 次印刷
字　　数	244 000	定　　价	14.90 元



面向21世纪课程教材



普通高等教育“九五”
国家教委重点教材

内 容 简 介

本书是教育部“高等教育面向21世纪教学内容和课程体系改革计划”的研究成果. 全书共讲述15个实验，它们分别为：微积分基础，怎样计算π，最佳分数近似值，数列与级数，素数，概率，几何变换，天体运动，迭代(一)——方程求解，寻优，最速降线，迭代(二)——分形，迭代(三)——混沌，密码，初等几何定理的机器证明. 最后还附有 Mathematica 简介.

本书适用于高等学校各专业本科生，以及具有初步的高等数学知识和计算机知识的其他读者.

前　言

本书是高等院校数学实验课的教材,适用于全国理、工、农、医、文各类高等院校,凡是开设高等数学课的学校都可以使用.

在大学中开设数学实验课,是教育部组织的"高等教育面向21世纪教学内容和课程体系改革计划"课题组的重要研究成果.该课程的教学对象,是全国所有高校,不分理工农医等科类的本科生.课程目的,是使学生掌握数学实验的基本思想和方法,即不把数学看成先验的逻辑体系,而是把它视为一门"实验科学",从问题出发,借助计算机,通过学生亲自设计和动手,体验解决问题的过程,从实验中去学习、探索和发现数学规律.

这是在我国高等学校中新开设的一门课程,还处于摸索和试点阶段.课程的指导思想、上课内容和方式都有待于在试点中逐步明确.该课程在数学专业和非数学专业都要开设,因此也是数学专业课题组和非数学专业课题组共同的任务.

中国科技大学数学系参加了数学专业和理科非数学专业面向21世纪教学内容和课程体系改革的两个课题组,并且受课题组委托进行了开设数学实验课的试点,产生了这本教材.参加这一试点工作的有我系的李尚志教授、陈发来教授、王树禾教授、侯定丕教授、吴耀华副教授、张韵华副教授等.为了真正检验我们的试点课的效果,看它是否能够引起学生的兴趣,是否能够达到预想的教学效果,是否适合于各不同专业的学生,我们采用了面向全校的选修课的形式,让全校所有专业的学生自愿自由选课.结果,在1998年和1999年的两次试点中,全校各专业选课的学生人数达到240人左右,几乎涵盖了所有的系科和专业,其中数学系学生约占1/3,非数学系的学生占2/3.学生们不但对于听课很有兴趣,而且主动积极地完成所布置的实验作业,在作业中表现出了很大的兴趣和创造性,还对开设这门课程的必要性和如何改进提出了很多很好的意见.

以下是我们对于开设数学实验课的想法和做法,也是这本书的主要指导思想.

一. 指导思想:

主要是让学生自己通过动手去体验,而不在于教完他们多少内容.不追求内容的系统性、完整性,而应当激发学生自己动手和探索的兴趣.

我们参照了物理实验课、化学实验课的内容和上课方法:这些实验课并不

需要花多少时间讲解理论和原理，讲解理论和原理是物理、化学的理论课程的任务．实验课主要是学生自己做实验，观察和分析实验结果．我们认为，数学实验也应当这样，不要在数学实验课上讲很多理论，也不应当花很多时间和精力来教算法．但是，物理、化学实验课往往是把实验的每个步骤都给学生规定得很详细，学生只要按步就班完成这些步骤，而实验的结果也是预先就知道了的，留给学生探索的余地不多．我们在设计数学实验课的时候就努力避免这种情况，尽量留些问题让学生自己去设计方法来解决，避免把实验课变成单纯传授计算技术的课程．

在设计数学实验内容的时候，虽然我们也有意识让学生通过实验学会一些基本的方法，但是我们并不以这些方法为线索组织课程内容．我们设计了一些能够引起学生兴趣的问题，每个实验围绕解决一个或几个问题来展开，教学生使用若干种方法来解决所给的问题，在解决问题中学习和熟悉这些方法，自己观察结果，得出结论．比如，围绕计算圆周率的近似值这一问题学习数值积分法、泰勒级数法、蒙特卡罗法、分数向无理数的最佳逼近；围绕光的折射定律和最速降线学习各种优化方法；围绕天体运动规律学习微分方程的数值解法；等等．

我们认为，尽管数学专业和非数学专业的学生的数学课程的难易程度有很大的差别，但数学实验课对他们来说却不必有多大的差别，基本的部分完全可以是共同的，只有一些理论较深的部分可以根据各自的情况有所取舍．我们在试点中的体会是，学生完成实验作业的难易主要不在于对数学知识的掌握程度，而是运用计算机能力的差别．我们感到，开设数学实验课以二年级为宜，让学生学过高等数学中必要的基本概念即可，不必学过很多的数学定理．这样，就可以有比较多的未知的东西供他们去探索．已学的东西太多，学生对探索的兴趣反而下降．实际上，选修我们课程的也有很多一年级学生，其中也有许多人的作业做得很好．因此，在一年级下学期开设这门课程也是可以的．

我们设想，数学实验可以包括两部分主要内容：第一部分是基础部分，围绕高等数学的基本内容，让学生充分利用计算机及软件的数值功能和图形功能展示基本概念与结论，去体验如何发现、总结和应用数学规律．另一部分是高级部分，以高等数学为中心向边缘学科发散，可涉及到微分几何，数值方法，数理统计，图论与组合，微分方程，运筹与优化等，也可涉及到现代新兴的学科和方向，如分形、混沌等．这部分的内容可以是新的，但不必强调完整性，教师介绍一点主要的思想，提出问题和任务，让学生尝试通过自己动手和观察实验结果去发现和总结其中的规律．即使总结不出来也没有关系，留待将来再学，有兴趣的可以自己去找参考书寻找答案．比如我们设计了一个实验让学生画分形的图形．这些图形的画法很简单，但画出来的结果却让学生感到神奇，引

起他们极大的兴趣,很多学生自己到图书馆找了有关分形的参考书来看.

二.关于数学实验课与一些相关课程的差别和联系,我们有如下看法:

与计算方法、统计方法、优化方法等课程的区别和联系:数学实验课尽管也要介绍和用到数值计算方法、统计方法、优化方法,但是不应取代这些课程.否则,学生会失去兴趣,认为反正还要上这些课程,何必上数学实验课呢?为划清这一界限,我们主张,数学实验课所用到的方法应当比较简单和浅显,可以由高等数学课程中的内容很快推出来,(其推导难度只应相当于高等数学作业题),而不需要花时间和精力作专门的讲解.而关于专门的、比较精细的专门方法的讲解,则留给这些课程去完成.当然,这些课程本身也应改革,不能纸上谈兵,也应有学生自己动手实践作为重要环节.

与数学建模课的区别和联系:数学建模与数学实验课都要用到计算机.但数学建模课是让学生学会利用数学知识和计算机手段来解决实际问题,而数学实验课侧重于在计算机的帮助下学习数学知识.一个是用,一个是学,两者的目标不同.从选材来说,我们主张两者都要从问题出发而不从概念出发.但数学建模强调问题的实用性而不强调普遍意义,解决问题本身就是目的;而数学实验课可以从理论问题出发,也可由实际问题出发,但这个理论问题或实际问题最好是比较经典的、具有普遍意义,让学生以解决问题为线索总结规律,学到知识.当然,数学实验课可以作为数学建模课的预备课程,使学生可以更快地掌握数学建模的基本方法和机能.

与高等数学课的区别和联系:都是为了学知识,但学习方法很不相同.高等数学课主要是由教师传授知识,而数学实验课则希望通过学生自己动手和观察去体会这些知识是怎样得出来的.

与计算机课程的联系:对于非计算机专业的学生来说,计算机知识(包括计算机语言以及软件的使用等)只是一种工具.好比学语言,不能只停留于学一个个的单字和一条条的语法规则,而必须通过阅读课文来学.并且,学了就要用,就要读报纸,读小说等.教计算机语言也是这样,应当结合解决一定的问题来学,学了就要用来解决问题,才有兴趣学,才能学得会,才不会忘记.但现在大学里的计算机语言课有时和其它课程(特别是数学课程)脱节,导致很多学生学习计算机语言也只是为了得学分,考过了就忘了,到高年级真需要用的时候还得重新复习.开设数学实验课有助于改变这种状况.数学实验课逼迫学生学好计算机知识来解决数学问题,数学实验课又为计算机课程提供了大量真刀真枪的练习机会.二者的结合势必真正提高学生对于计算机知识的掌握水平和应用的能力,使计算机课程可以用更少的课时取得更好的效果.

数学实验课的开设,不但本身就是高等数学课程体系的一项改革,而且必将促进其它许多相关课程的改革.

三. 使用本书的注意事项：

本书可以说是像《聊斋》而不像《三国演义》，不强调系统性，各个实验基本上是独立的. 后面的实验一般不会用到前面的实验的知识，(但也有几个例外). 实验安排的顺序基本上是由易到难，基础内容在前，较高级的扩散性的内容在后(但也不是绝对的). 虽然内容的编排实际上还是有一定的系统性，但我们并不强调这一系统性. 我们强调的是从问题出发，自己动手做，自己观察结果，并且鼓励学生自己发明出新的实验.

因此，在使用本书时，不必按顺序从头到尾依次做各个实验，更不必全部做. 可以根据各学校和学生的情况选做其中一部分实验. 我们自己的做法是：每两周做一个实验，整个课程做 8 个到 9 个实验. 也可以老师讲 8 到 9 个实验，但只要求学生在一学期中完成 4 个到 5 个实验，并且欢迎他们以后自己完成更多的实验. 有些学校暂时没有时间完整开设数学实验课，可以先在高等数学课中结合课程内容安排做某些实验，甚至可以在计算机课程中作为作业安排做某些实验. 总之，只要认真去做，做一个实验就有一分收获.

在每次实验中，我们的作法是先由教师讲两个课时，主要是提出问题，适当介绍问题的背景，介绍主要的实验原理和方法. 然后就让学生自己动手去做，自己去折腾，去观察，通过观察得出结论. 本来，实验结果一般都可以用理论推导出来，但这绝不是本课程的目的，教师千万不要花很多时间去作这种理论推导，最好也不要预先告诉学生实验的结果，实验结果让学生自己去观察得出. 本书在某些实验后面作为附录作了这种推导，只是为了供有兴趣的学生课外阅读之用，决不是供教师讲解的教材，也不作为课程要求和考察内容，更不要以考试为手段强迫那些不感兴趣的学生去弄懂. 在我们的试点中，有的学生希望少讲一些理论，多给他们留一些自己探索的空间，也有的学生希望多讲一些理论. 我们说：能够通过实验使学生希望多学理论，这就是好事情. 将他们学习数学理论的兴趣激发起来了，胃口吊起来了，这门课的目的就达到了. 数学实验是“开胃汤”，而不是大餐. 胃口吊起来之后希望多“吃”一些，请到正规的“餐馆”去，学生可以通过看参考书和学习其它课程来满足对于理论学习的渴求.

本书各实验大体上以 Mathematica 软件作为主要使用的收件，并至在一些实验后面附上了 Mathematica 程序供参考，以避免学生或老师在计算机方面花费太多的时间，冲淡了探索数学内容的趣味. 但这并不意味着本书的实验只能用 Mathematica 去做，更不意味着没有 Mathematica 就不能做这些实验. 实际上，课本中对于计算机的要求也是相当开放的，实验方法的叙述和建议着重于数学思路，这些思路可以用 Mathematica 来实现，也可以用别的软件或自己编程完成.

我们的试点课没有专门的考试，评定成绩的唯一依据是平时的实验报告.实验报告的评分的最基本标准是要自己动手，要写上自己观察到的现象并进行分析.实话实说，不能造假.哪怕观察到的现象与预计不一致，或者与理论推导的结果不一致，也不能在实验报告中说假话，而应当分析其原因，找出改进的办法，重做实验，重新得出结论.对实验报告的更高的标准是创造性.对于有创造性的报告，要给以高分作为鼓励.教师批改了实验报告之后，要在下一次实验开始时，对以前的实验中出现的优点和缺点进行评讲，包括让学生参加讨论.

以上是我们对于开设数学实验课的想法和作法，本书则是这些想法和做法的物化和固化.数学实验课的开设在我国还是一件新的事情，处于摸索阶段.怎样开设，势必有许多不同的想法和作法.我们认为，现阶段应当鼓励各种不同的想法和作法，各自进行自己的探索和试点.可以而且应当相互交流，但不必统一，也不必争论哪种做法更好.现在首先是要先干起来，经过若干年实践去积累和总结经验，根据实践的效果来逐渐完善和成熟.

由于时间比较仓促，我们自己的试点还很不完善，本书的写作中的不完善和疏漏之处一定不少.现在的这本书只能算个毛坯，先供自己和大家使用起来，在使用中改进.希望在不久的将来能将本书改得更好.

感谢两个教改课题组的负责人萧树铁教授和姜伯驹教授，正是在他们的领导下才使我们开始了数学实验的教改试点，才有可能产生这本书.数学实验课程试点的成功，还离不开中国科技大学和数学系的领导的支持，以及学校各部门、特别是教务部门的支持，他们对教育改革的一贯支持，以及在全校范围内造成的搞教改的热烈而宽松的气氛，是我们作出成绩的重要保证.感谢数学系王树禾教授、侯定丕教授参加试点和讲课，感谢选修这门课程的广大学生，正是他们的积极认真地参与，才使得我们开设数学实验课的试点搞得有声有色，使本书的指导思想和内容得以形成和完善.

在本书的写作过程中，我们参考了由 Mount Holyoke College（美）编写的、清华大学白峰杉和蔡大用翻译的、高等教育出版社与施普林格出版社合作出版的《数学实验室》一书，并且受到启发.在此感谢该书的作者、译者和出版者.

作　者

1999 年 6 月于合肥

责任编辑 邵 勇
封面设计 张 楠
责任绘图 孟庆祥 陈钧元
版式设计 邵 勇
责任校对 李 陶
责任印制 杨 明

目　　录

实验一　微积分基础

本实验的目的是:学习使用 Mathematica 的一些基本功能来验证或观察得出微积分学的几个基本结论.

Mathematica 是一个符号计算系统.它可以帮助你很容易进行初等数学和高等数学中的数值计算、符号计算、画图等各种事情.

1.1　函数及其图象

Mathematica 画区间 $[a,b]$ 上的函数 $y=f(x)$ 的图象的语句的格式是:

```
Plot[f,{x,a,b}]
```

画参数方程 $x=f(t),y=g(t)$,$t\in[a,b]$ 所表述的曲线的语句的格式是:

```
ParametricPlot[{f,g},{t,a,b}]
```

练习 1

(1)在同一坐标系里作出函数 $y=x^3-6x+2$ 及其导函数 $y'=3x^2-6$ 的图象.Mathematica 语句如下:

```
Plot[x^3 - 6 x + 2,3 x^2 - 6,{x, - 5,5}]
```

注:这里画的是区间 $[-5,5]$ 上的图象.你也可选别的区间试试.

观察以下现象:

(i)当 $y'>0,<0$ 时 y 的图象的升降情况.当 $y'=0$ 时 y 是否有极大值或极小值?

(ii)当 y' 上升或下降时 y 的图象的凸凹情况.当 y' 取极值时 y 的图象是

否出现拐点？

(iii)观察得出方程 $y=0$ 的根的近似值 a，比如 $a=2$．再用以下语句求 2 附近的根的更精确的值：

```
FindRoot[x^3 - 6 x + 2,{x,2}]
```

求根的原理是：将函数 $y=f(x)$ 在 $x=a$ 附近看作一次函数 $y\approx f(a)+f'(a)(x-a)$，其中 $f'(a)=3a^2-6$ 是 y 在 $x=a$ 处的导数值．认为一次方程 $f(a)+f'(a)(x-a)=0$ 的解

$$a_1=a-\frac{f(a)}{f'(a)}$$

是比 a 更好的近似值．用它代替 a 再求出根的更好的近似值．这个方法叫做**牛顿切线法**．本例中由 a 求 a_1 的过程可以用下面的语句来定义：

```
g[a_]:= a - (a^3 - 6 a + 2)/(3 a^2 - 6)
```

这里在定义函数 g 时用 a_ 代表变量 a 是等待在应用时将 a 代入具体值．从初始值 2 出发用函数 g 求根时，先求 $g(2)$，再求 $g(g(2))$，再求 $g(g(g(2)))$，…．直到得到一个数 a_0 使 $g(a_0)=a_0$，即 $f(a_0)=0$，a_0 就是所求的根．在 Mathematica 中，将函数 g 对 2 连续作用若干次（比如作用 4 次），并且列出每次作用的结果，可以用下面的语句实现

```
NestList[g,2,4]
```

(iv)观察得出使函数 y 取极大或极小值的 x 值的近似值 x_0，再用以下语句求 x_0 附近的极大或极小值点的更精确的值．

```
FindMinimum[x^3 - 6 x + 2,{x,x0}]
```

另选函数 $y=f(x)$ 研究上述问题．

(2) 分别作出函数 $y=\sin\frac{1}{x}$ 在区间 $[-1,1]$，$[-0.1,0.1]$，$[-0.01,0.01]$ 的图象，观察图象在 $x=0$ 附近的形状．

在同一坐标系中作出点集 $t=\{(1/k,\sin k)\mid 1\leqslant k\leqslant 2\,000\}$，观看点集中隐藏了什么图象．Mathematica 作图语句为：

```
t = Table[{1/k,Sin[k]},{k,1,2000}]  (* 定义点集 t.)
ListPlot[t]  (* 画出点集 t 中所有的点.)
ListPlot[t,PlotJoined->True]  (* 画出 t 中的点依次连接成的光滑曲线.)
```

注：圆括号中星号 * 后面是对语句作用的说明.

(3)作出函数 $y = \frac{\sin x}{x}$ 在区间 $[-0.1, 0.1]$ 上的图象，观察图象在 $x = 0$ 附近的形状.

练习 2

级数与无穷乘积.

(1)在同一坐标系里作出函数 $y = \sin x$ 和它的 Taylor 展开式的前几项构成的多项式函数 $y = x - \frac{x^3}{3!}, y = x - \frac{x^3}{3!} + \frac{x^5}{5!}, \cdots$ 的图象. 观察这些多项式函数的图象向 $y = \sin x$ 的图象逼近的情况.

(2)分别取 $n = 10, 20$，画出函数 $y = \sum_{k=1}^{n} \frac{1}{2k-1}\sin(2k-1)x$ 在区间 $[-3\pi, 3\pi]$ 上的图象. 当 $n \to \infty$ 时，这个函数趋向于什么函数？

(3)分别取 $n = 5, 10, 15$，在同一坐标系里作出函数 $f(x) = \sin x$ 与 $p(x) = x \cdot \prod_{k=1}^{n}\left(1 - \frac{x^2}{k^2\pi^2}\right)$ 在区间 $[-2\pi, 2\pi]$ 上的图象，观察当 n 增加时 $p(x)$ 的图象向 $\sin x$ 的图象逼近的现象. 注意函数 $p(x)$ 的根 $\pm k\pi$ ($k = 0, 1, 2, \cdots, n$)也都是 $f(x)$ 的根，且两个函数在 $x = 0$ 处的导数相同. 在任何有限区间上，当 $n \to \infty$ 时函数 $p(x)$ 逼近 $\sin x$.

完成练习 2 的三项作图任务的 Mathematica 语句分别是：

```
s[x_,n_]:= Sum[(-1)^(k-1)x^(2k-1)/((2k-1)!),
              {k,1,n}]
Plot[{Sin[x],s[x,2],s[x,3],s[x,4]},{x,-2Pi,2Pi}]
f[x_,n_]:= Sum[Sin[k*x]/k,{k,1,n,2}]
Plot[f[x,10],{x,-2Pi,2Pi}]
p[x_,n_]:= x*Product[1-x^2/(k^2Pi^2),{k,1,n}]
Plot[{Sin[x],p[x,5]},{x,-2Pi,2Pi}]
```

以上语句中的 s[x,2]… 等可以换成 s[x,5]… 等，f[x,10]，p[x,5] 可以换成 f[x,20]，p[x,10]，p[x,15] 等.

1.2 数 e

中学学的对数以 10 为底，称为常用对数，记作 $\lg N$. 但科学上常用的对数却以一个无理数 $e = 2.71828\cdots$ 为底，称为自然对数，记作 $\ln N$ 或 $\log N$. 为什么以这个稀奇古怪的无理数 e 为底的对数反而比以 10 为底的对数更自然？

公元 17 世纪纳皮尔（J. Napier）发明对数时，其目的是简化天文数据的计算，将乘法转化为加法来计算. 它希望将每个正实数 N 表示为某个给定的正实数 a 的幂：$N = a^n$. 如果 $N = a^n, M = a^m$ ，则 $M \cdot N = a^{m+n}$ ，M, N 的乘法变成了 m, n 的加法. 问题是必须预先编制一个表，也就是对数表，列出幂（即真数）N 与指数（即对数）n 之间的对应关系. 如果以 10 为底，则对数表是下面的样子：

幂（真数）	1	10	100	1 000	…
指数（对数）	0	1	2	3	…

表中的真数跳跃太大. 比如，1 过了就是 10，而它们之间的 2, 3, …, 9 都没有. 以后从10跳到100，从 100 跳到1 000，跨度就更大了. 要克服这个缺点，使表中相邻两个真数比较接近，就应当取底 a 接近于 1. 比如取 $a = 1.001$（纳皮尔取 $a = 0.99999$，以便于计算三角函数）. 如果你愿意，可以自己编制一个以 $a = 1.001$ 为底的对数表，列出以 $n = 1, 2, \cdots, 1000$ 为指数的所有的幂. 如果你嫌这个表太长了，可以不列出这样多的幂，而按下面的练习去做.

练习 3

(1)编制以 $a = 1.001$ 为底的对数表，在表中列出 a 的 1, 2, 3, …, 9, 10, 20, …, 90, 100, 200, …, 900, 1 000, 2 000, …, 9 000, 10 000 次幂.

利用你编制这个表可对满足条件 $1 \leqslant N \leqslant a^{10000}$ 的实数 N 求出 $b = \log_a N$. 方法如下：先查出使 $N_3 = a^{1000n_3} \leqslant N$ 的最大整数 n_3；计算 $\lambda_2 = N/N_3$，查出使 $N_2 = a^{100n_2} \leqslant \lambda_2$ 的最大整数 n_2；计算 $\lambda_1 = \lambda_2/N_2$，查出使 $N_1 = a^{10n_1} \leqslant \lambda_1$ 的最大整数 n_1；计算 $\lambda_0 = \lambda_1/N_1$，查出使 $N_0 = a^{n_1} \leqslant \lambda_0$ 的最大整数 n_0. 则 $b_0 = n_0 + 10n_1 + 100n_2 + 1000n_3$ 是 $b = \log_a N$ 的不足近似值，误差 < 1. 还可以稍微估计一下 b 的小数部分 $b - b_0$. 为此，计算 $\lambda_{-1} =$

λ_0/N_0，则 $1 \leqslant \lambda_{-1} < a = 1.001$．在很小的区间 $[0, 0.001]$ 内近似地将指数函数 $y = f(x) = 1.001^x$ 看作一次函数，则由 $f(0) = 1$，$f(1) = 1.001$ 可得 $f(x) \approx 1 + 0.001x$．解方程 $\lambda_{-1} = 1.001^x \approx 1 + 0.001x$ 可得近似解 $x \approx (\lambda_{-1} - 1)/0.001$．于是 $b \approx b_0 + (\lambda_{-1} - 1)/0.001$．

利用你编制的对数表，用上述方法求 $b_1 = \log_a 2$ 和 $b_2 = \log_a 10$ 的近似值，由此求出 $\lg 2$ 的近似值 b_1/b_2．

不难看出，用接近于 1 的 $a = 1.001$ 为底编制对数表要比以 10 为底优越．唯一的美中不足是求出来的对数的数值都太大．这个缺点很容易改正：只要将所有的对数缩小同一个倍数就行了．鉴于 $a = 1 + 1/1\,000$，很自然考虑将所有的对数除以 1 000，取 $0.001\log_{1.001}N$ 代替 $\log_{1.001}N$．这样一来，原先对数为1 000的数 $a_3 = 1.001^{1\,000}$ 的对数变为 1，而 $0.001\log_{1.001}N$ 恰是以 a_3 为底的对数．相当于一开始就取了 $a_3 = 1.001^{1\,000}$ 来作为对数的底．如果要提高精确度，可以取更接近 1 的1.000 1来代替1.001，也就是取 $a_4 = 1.000\,1^{10\,000}$ 来作为对数的底，还可以进一步取 $a_5 = 1.000\,01^{100\,000}$ 作为对数的底．一般，可以考虑 $a_n = (1 + 1/n)^n$ 作为对数的底，n 越大越好．

练习 4

观察当 n 趋于无穷大时数列 $a_n = (1 + 1/n)^n$ 和 $A_n = (1 + 1/n)^{n+1}$ 的变化趋势：

(1)求出当 $n = 10^m$，$m = 1, 2, \cdots, 7$ 时 a_n, A_n 的值．观察变化趋势．

计算 a_n 的 Mathematica 语句为

```
Do[Print[(1 + 10^(-m))^(10^m)],{m,1,7}]
```

计算 A_n 的语句类似．

(2)在同一坐标系中画出下面三个函数的图象：

$$y = (1 + 1/10^x)^{10^x}, \quad y = (1 + 1/10^x)^{10^x + 1}, \quad y = \mathrm{e}$$

观察当 x 增大时图象的走向．

Mathematica 画图语句为：

```
Plot[{(1 + 10^(-x))^(10^x),
        (1 + 10^(-x))^(10^x + 1),E},{x,1,4}]
```

这个语句画的是函数在区间 [1,4] 内的图象. 再将区间换成 [2,4] 或 [3,5] 或 [5,6] ,观察所得的图象.

通过观察可以看到,当 n 增大时 $a_n=(1+1/n)^n$ 递增, $A_n=(1+1/n)^{n+1}$ 递减. 随着 n 的无穷增大, a_n,A_n 无限接近,趋于共同的极限 e = 2.718 28⋯ . 以这个 e 为底的对数称为自然对数.(当然,这些结论都可以通过理论推导加以证明,请读者自己参考微积分教材,这里就不重复了.)

(3)用下面的语句计算 $e=1+\sum_{k=1}^{\infty}\frac{1}{k!}$ 的近似值,精确到小数点后面 15 位.

```
Do[Print[N[1.0 + NSum[1/(k!),{k,1,n}],15],{n,5,20}]
```

(* 其中语句 **N[表达式,15]**表示以 **15** 位精度的实数输出表这式,语句 **NSum[表达式,{k,1,n}]**表示 $\sum_{k=1}^{n}$ (表达式).)

上面是通过对数表的编制来说明自然对数是怎样自然产生的. 虽然当初纳皮尔编制对数表的时候还没有这样明确地提出自然对数,但他一开始编制的决不是以 10 为底的常用对数表,他以0.999 99为底编制的对数表从本质上接近于自然对数表. 只是到后来,为了使用的方便,才采用换底公式将已编成的对数表改成了以 10 为底的常用对数表.

在科学中广泛应用以 e 为底的自然对数的更直接的理由是:它使涉及到对数的微分和积分公式变得最为简单.

练习 5

(1)运行下面 Mathematica 语句,计算当 $x=10^{-n}$, $n=1,2,3,4,5,6,7$ 时, $\lambda(x)=(\lg(1+x))/x$ 的值.

```
Do[Print[Log[10,1.0 + 10.0^(-n)]/(10^(-n))],{n,1,7}]
```

(* 在 Mathematica 中,**Log[a,N]**表示以 **a** 为底的 **N** 的对数,从而 **Log[10,N]**表示常用对数 $\lg N$. 而**Log[N]**表示自然对数,即以 e 为底的对数,此时将 e 略去不写. 循环语句"**Do[循环体,{n,1,7}]**"表示对 **n = 1,2,⋯,7** 依次执行循环体. 本例的循环体为"**Print[表达式]**",其作用是将表达式的值印出来.)

观察当 x 趋于 0 时 $\lambda(x)$ 是否趋近于某一个极限值 λ.

λ 就是常用对数 $y = \lg x$ 在 $x = 1$ 处的导数. 它不是一个简单的数. 定义 $y = f(x) = \lambda^{-1}\lg x$，则 $f(x)$ 在 $x = 1$ 处的导数

$$\lim_{\delta\to 0}\frac{f(1+\delta)-f(1)}{\delta} = 1$$

而 $f(x) = \lg x/\lg(10^{\lambda}) = \log_a x$ 是以 $a = 10^{\lambda}$ 为底的对数.

(2)计算 $a = 10^{\lambda}$.

(3)计算当 $x = 10^{-n}$，$n = 1,2,3,4,5,6,7$ 时，$\mu(x) = (\ln(1+x))/x$ 的值. Mathematica 语句为

```
Do[Print[Log[1.0 + 10.0^(-n)]/(10^(-n))],{n,1,7}]
```

观察当 x 趋于 0 时 $\mu(x)$ 是否趋于某一个极限值 μ？μ 的值是多少？

上面主要研究的是与 e 有关的极限. 请自己用类似方法研究其它一些极限.

1.3 积分与自然对数

对正实数 a，研究反比例函数 $y = f(x) = 1/x$ 的图象与 x 轴及直线 $x = 1$，$x = a$ 所围的面积 $S(a)$，即定积分 $\int_1^a \frac{1}{x}dx$. 注意，我们定义直线 $x = 1$ 为计算面积的基准，从这条直线向右的面积为正，向左的面积为负. 也就是当 $a > 1$ 时 $S(a) > 0$，当 $0 < a < 1$ 时 $S(a) < 0$，而当 $a = 1$ 时 $S(a) = 0$.

以 $a = 2$ 的情形为例. 将自变量区间 $[1,2]$ 平均分为 n 等份，每份长度为 $1/n$，插入的分点 $x_k = 1 + k/n$ ($1 \leqslant k \leqslant n-1$). 所求面积 $S(2)$ 相应地被分成 n 份，其中第 k 份是直线 $x = x_{k-1}$，$x = x_k$ 之间所夹的部分，其面积记为 S_k . S_k 的上边界是曲线 $y = 1/x$ 在区间 $[x_{k-1}, x_k]$ 上的一段. 最大高度为 $1/x_{k-1} = n/(n+k-1)$，最小高度为 $1/x_k = n/(n+k)$. 故

$$\frac{1}{n}\frac{n}{n+k-1} = \frac{1}{n+k-1} > S_k > \frac{1}{n}\frac{n}{n+k} = \frac{1}{n+k}$$

于是

$$\Sigma_n = \sum_{k=1}^{n} \frac{1}{n+k-1} > S(2) > \sigma_n = \sum_{k=1}^{n} \frac{1}{n+k}$$

Σ_n,σ_n 分别称为“大和”、“小和”，而面积的准确值 $S(2)$ 在它们之间. 当 n 递增时，Σ_n 递减，σ_n 递增，当 $n\to\infty$ 时二者趋于共同的极限 $S(2)$. 易见 $\Sigma_n-\sigma_n=1/2n$ ，用 Σ_n,σ_n 来作 $S(2)$ 的近似值，误差小于 $1/2n$. 而它们的平均值 $(\Sigma_n+\sigma_n)/2$ 与 $S(2)$ 的误差则应当更小. 事实上，这个平均值就是用梯形公式求出的 $S(2)$ 的近似值.

练习 6

(1)对 $n=10^m$ ($m=3,4,5,6$)，用 **NSum** 语句计算“大和” Σ_n 和“小和” σ_n 及它们的平均值 $(\Sigma_n+\sigma_n)/2$ ，观察它们的变化趋势. 得出 $S(2)$ 的近似值. 再用求数值积分的语句

```
NIntegrate[1/x,{x,1,2}]
```

求 $S(2)$ ，将所得的结果与前面用定义求出的结果相比较.

(2)用语句

```
S[x_]:= NIntegrate[1/t,{t,1,x}]
```

定义函数 $S(x)=\int_1^x \frac{1}{t}\mathrm{d}t$. 再用画图语句

```
Plot[S[x],{x,0.1,5}]
```

画出这个函数在区间 [0.1,5] 上的图象(也可以换成别的区间). 观察图象的形状，它好象是什么函数?

(3)观察出(2)中画出的图象可能是对数函数的图象. 现在来求这个对数的底 b ，它应满足条件 $S(b)=1$. 从图象上看 b 比 3 稍小一些. 以 3 作为初始值，用练习 2 所介绍的 **FindRoot** 语句求 b 所满足的方程 $S(x)-1=0$ 的根的较精确的值. 也可以用练习 2 介绍的牛顿切线法，利用递推关系式

$$a_1 = a - \frac{S(a)-1}{S'(a)}$$

由根的近似 a 求出更好的近似值 a_1 . 这里 $S'(a)=1/a$. 迭代语句为

```
g[a_]:= a - (S[a] - 1)a
NestList[g,3,4]
```

计算结果发现 b 恰是自然对数的底 e．在同一坐标系中作出函数 $S(x)$ 和 $\ln x$ 的图象，发现二者完全重合．

当然，由不定积分很容易知道 $S(x)=\int_1^x \frac{1}{t}\mathrm{d}t=\ln x$．以上是通过数值和图象实验来验证这一结论．

Mathematica 求不定积分 $\int f(x)\mathrm{d}x$，定积分 $\int_a^b f(x)\mathrm{d}x$ 以及用数值方法求定积分 $\int_a^b f(x)\mathrm{d}x$ 的语句分别是

```
Integrate[f,x]
Integrate[f,{x,a,b}]
NIntegrate[f,{x,a,b}]
```

1.4 调和数列

自然数的倒数组成的数列 $1,\frac{1}{2},\frac{1}{3},\cdots,\frac{1}{n},\cdots$ 称为调和数列．我们把它的前 n 项和 $\sum_{k=1}^{n}\frac{1}{k}$ 记作 $H(n)$．

练习 7

(1)将坐标为 $(n,H(n))$ ($n=1,2,\cdots,100$)的点依次连接成光滑曲线．与自然对数的曲线 $y=\ln x$ ($x\in[1,100]$)画在同一个坐标系中．完成此事的 Mathematica 语句如下：

H[n_]:= NSum[1/k,{k,1,n}] （* 定义函数 H(n).）

t = Table[{n,H[n]},{n,1,100}] （* 定义点集 t.）

pic1 = ListPlot[t,PlotJoined -> True] （* 将 t 中的点顺次连接成光滑曲线并储存在 pic1 中（以便重画），其中的选项“PlotJoined -> True”表示顺次连成光滑曲线．如果去掉此选项，则只将 t 中的点画出．）

pic2 = Plot[Log[x],{x,1,100},PlotStyle -> {RGB[0,0,1]}] （*

画函数 $y = \ln x (x \in [1,100])$ 的图象并储存在 **pic2** 中. 选项"**PlotStyle -> {RGB[0,0,1]}**"是将此曲线画成蓝色,函数 **RGB[r,g,b]** 中的三个自变量 **r, g, b** 取值在 **0** 到 **1** 之间,分别表示红、绿、蓝三种颜色的强度.)

```
Show[pic1,pic2]  (* 将 pic1,pic2 表示的两个图形同时画出.)
```

观察可知当 n 很大时同一 n 的 $H(n)$ 与 $\ln n$ 之差接近于常数. 计算出 $c = H(100) - \ln 100$, 用红色画出函数 $y = \ln x + c$ 的图象, 并与 **pic1**, **pic2** 的图象画在同一个坐标系中. 语句如下:

```
c = H[100] - Log[100]
pic3 = Plot[Log[x] + c,{x,1,100},PlotStyle -> {RGB[1,0,0]}]
Show[pic1,pic2,pic3]
```

观察 **pic1** 的黑色曲线与 **pic3** 的红色曲线几乎重合的现象.

(2)为了研究当 n 无穷增大时 $C(n) = H(n) - \ln n$ 是否趋孟一个常数, 将坐标为 $(n, H(n) - \ln n)$ ($1 \leqslant n \leqslant 100$)的点依次连接成光滑曲线 c1, 再将坐标为 $(n, H(n) - \ln(n+1))$ ($1 \leqslant n \leqslant 100$)的点依次连接成光滑曲线 c2. 在同一坐标系中画出曲线 c1, c2. 观察 c1 递减和 c2 递增以及二者相互接近的现象.

计算 $n = 10^m$ ($m = 1, 2, \cdots, 6$) 时 $C(n) = H(n) - \ln n$ 和 $c(n) = H(n) - \ln(n+1)$ 的值. 观察 $C(n)$ 递减、$c(n)$ 递增、二者趋于同一极限的现象. 并求出这个极限 C.

极限

$$C = \lim_{n \to \infty}\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} - \ln n\right)$$

称为**欧拉(Euler)常数**.

显然

$$c(n) = H(n) - \ln(n+1) < C(n) = H(n) - \ln n$$

$c(n)$ 就是下页上图中前 n 个阴影"三角形"之和, 当然随着 n 的增加而递增.

而 $C(n)$ 就是单位正方形面积 1 减去下页下图中前 n 个阴影"三角形"面积所得的差, 当然随着 n 的增加而递减.

而由于当 $n \to \infty$ 时 $C(n) - c(n) = \ln\left(1 + \frac{1}{n}\right)$ 趋于 0, 故 $C(n), c(n)$ 趋于同一个极限 C.

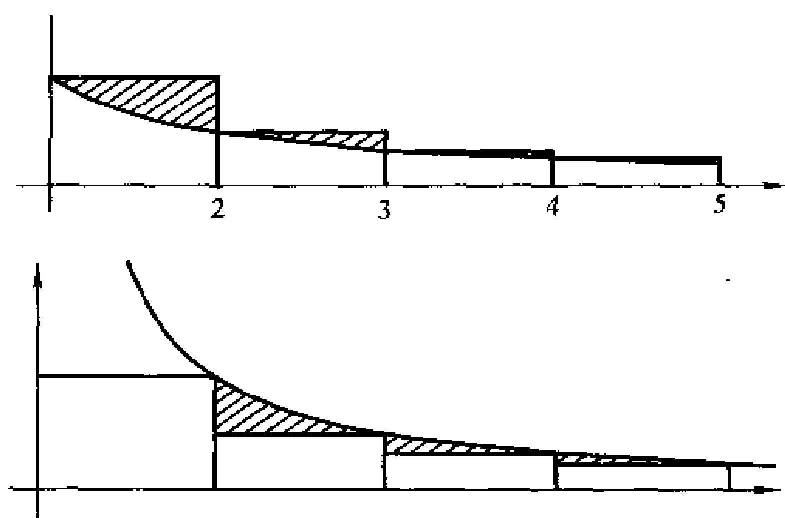


图 1—1

1.5 双曲函数

设 $P(a,b)$ 是单位圆 $x^2+y^2=1$ 上一点，$t=\angle XOP$，则定义

$$a=\cos t,\quad b=\sin t$$

分别称为 t 的**正弦和余弦**.

自变量 t 也可用另一方法定义：设 $\angle XOP$ 被圆截得的扇形（图中阴影部分）面积为 S，且规定 S 的正负号与 $\angle XOP$ 相同. 则 $S=t/2$，从而 $t=2S$.

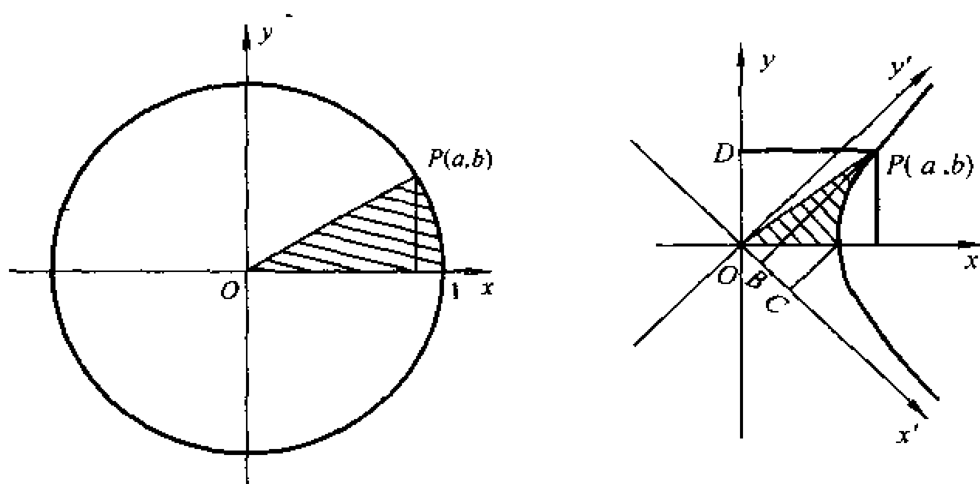


图 1—2

现在将圆换成双曲线 $x^2 - y^2 = 1$. $P(a,b)$ 是双曲线右边一支上任一点，$a \geqslant 1$. 设 $\angle XOP$ 被双曲线截得的曲边三角形(图中阴影部分) AOP 面积为 S. 且规定 S 的正负号与 $\angle XOP$ 相同. 记 $t = 2S$. 定义

$$a = \mathrm{ch}\, t, \quad b = \mathrm{sh}\, t$$

分别称为 t 的双曲**正弦**和**双曲余弦**.

求 S 的方法：记双曲线的右边一支与 x 轴的交点为 $A(1,0)$ ，过 P 点与 x 轴平行的直线与 y 轴的交点为 $D(0,b)$. 不妨将 y 看作自变量，$x = \sqrt{y^2+1}$ 看作因变量. 则双曲线弧 AP 与直线段 PD,OA,OD 所围的曲边梯形 $AODP$ 的面积 $S_{AODP} = \int_0^b \sqrt{y^2+1}\,\mathrm{d}y$. 所以

$$S = S_{AODP} - S_{\triangle OPD}$$
$$= \int_0^b \sqrt{y^2+1}\,\mathrm{d}y - \frac{1}{2}b\sqrt{b^2+1}$$
$$t = 2\int_0^b \sqrt{y^2+1}\,\mathrm{d}y - b\sqrt{b^2+1}$$

练习 8

画出双曲函数 $y = \mathrm{ch}\, t$ 和 $y = \mathrm{sh}\, t$ 的图象. Mathematica 语句如下：

```
S[y_]:= 2*NIntegrate[Sqrt[u^2+1],{u,0,y}]
                        -y*Sqrt[y^2+1]
sh = ParametricPlot[{t[y],y},{y,-5,5}]
ch = ParametricPlot[{t[y],Sqrt[y^2+1]},{y,-5,5}]
```

观察发现双曲余弦的图象有些象抛物线. 它通过点 $(0,1)$ 并且以 y 轴为对称轴. 如果它是抛物线，应具有形状 $y = ax^2 + 1$. 任取一个 u 值，可决定 $y = \mathrm{ch}\, t$ 上一点的坐标 $(t(u), \sqrt{u^2+1})$. 取 $a = \dfrac{\sqrt{u^2+1}-1}{t^2(u)}$ ，则抛物线 $y = ax^2 + 1$ 与 $y = \mathrm{ch}\, t$ 的图象有三个公共点：$(0,1)$，$(\pm t(u), \sqrt{u^2+1})$. 如果 $y = \mathrm{ch}\, t$ 是抛物线，则它与抛物线 $y = ax^2 + 1$ 应重合.

取 $u = 1.5$ ，计算出 a ，在同一坐标系中画出抛物线 $y = ax^2 + 1$ 与 $y = \mathrm{ch}\, t$ 的图象. (不妨将抛物线画成红色，以便辨认.) 可以观察到，虽然两条曲线有三个公共点，但二者并不重合. 随着 $|x|$ 的增大，$y = \mathrm{ch}\, t$ 的图象比抛物线更快地向 y 轴靠拢.

取 $u=0.1,0.01,0.001,\cdots$ 计算出相应的 a 值，观察当 $u\to 0$ 时 a 的极限值 a_0. 在同一坐标系中作出 $y=a_0x^2+1$ 与 $y=\operatorname{ch} t$ 的图象，观察发现二者在点 $(0,1)$ 的附近吻合得很好. 当抛物线 $y=ax^2+1$ 与 $y=\operatorname{ch} t$ 的公共点 $(\pm t(u),\ \sqrt{u^2+1})$ 的位置趋于 $(0,1)$ 时，该抛物线的极限位置就是 $y=a_0x^2+1$. $1+a_0x^2$ 也就是 $y=\operatorname{ch} t$ 的泰勒展开式的前三项(一次项等于 0).

$y=\operatorname{ch} t$ 的图象称为**悬链线**. 它就是将一条绝对柔软、无伸缩性、密度均匀的细线两端固定后自然下垂所成的形状.

前面求与点 $P(a,b)$ 对应的 $S=S_{AOP}$ 及 $t=2S$ 还有另外一种方法：将坐标轴向顺时针方向旋转 45°，得到新的坐标系 $Ox'y'$. 由坐标变换公式，同一点的旧坐标 (x,y) 与新坐标 (x',y') 之间有下面的关系式：

$$x=\frac{x'+y'}{\sqrt{2}},\quad y=\frac{x'-y'}{\sqrt{2}}$$

从而双曲线方程 $x^2-y^2=1$ 变为 $2x'y'=1$. $P(a,b)$ 的新坐标 (a',b') 为 $a'=\dfrac{a-b}{\sqrt{2}},b'=\dfrac{a+b}{\sqrt{2}}$，$A$ 的新坐标为 $\left(\dfrac{1}{\sqrt{2}},\dfrac{1}{\sqrt{2}}\right)$. 分别过 P,A 作 Ox' 轴的垂线交 Ox' 轴于 B,C. 则 $OB=a'$,$OC=\dfrac{1}{\sqrt{2}}$. 由于 P,A 都在双曲线上，故 $a'b'=\left(\dfrac{1}{\sqrt{2}}\right)\left(\dfrac{1}{\sqrt{2}}\right)$，即 $OB\cdot BP=OC\cdot CA$，$S_{\triangle BOP}-S_{\triangle COA}=\dfrac{OB\cdot BP}{2}-\dfrac{OC\cdot CA}{2}=0$. 于是

$$S=S_{ACBP}+S_{\triangle BOP}-S_{\triangle COA}=\int_{a'}^{\frac{1}{\sqrt{2}}}\frac{1}{2x'}\mathrm{d}x'$$

$$=\frac{1}{2}\left(\ln\frac{1}{\sqrt{2}}-\ln\frac{a-b}{\sqrt{2}}\right)=\frac{1}{2}\ln\frac{1}{a-b}=\frac{1}{2}\ln(a+b)$$

$$t=\ln(a+b)=\ln(\sqrt{b^2+1}+b)$$

注意，上面用到了双曲线上任一点 $P(a,b)$ 的坐标所满足的关系式 $a^2-b^2=1$，$1/(a-b)=a+b$. 由 $t=\ln(a+b)$ 及 $-t=\ln[(a+b)^{-1}]=\ln(a-b)$ 可得 $\mathrm{e}^t=a+b$,$\mathrm{e}^{-t}=a-b$，从而

$$a=\operatorname{ch} t=\frac{\mathrm{e}^t+\mathrm{e}^{-t}}{2},\quad b=\operatorname{sh} t=\frac{\mathrm{e}^t-\mathrm{e}^{-t}}{2}$$

这就是课本上对双曲函数的定义. 以上介绍的从与双曲线有关的面积来定义 $\operatorname{ch} t$,$\operatorname{sh} t$，说明了为什么要将这两个函数称为双曲函数，以及它们与三角函数的类似之处.

实验二　怎样计算 π

众所周知，圆周率 π 是平面上圆的周长与直径之比，它等于 3.141 592 6…. 古代人把 3 作为它的近似值. 古希腊阿基米德(Archimedes)曾得到 $3\frac{10}{71} < \pi < 3\frac{1}{7}$. 我国宋代的祖冲之得到 π 的近似值 $\frac{22}{7}$（约率）和 $\frac{355}{113}$（密率），后者化为小数后等于 3.141 592…，与 π 的准确值的误差在 10^{-6} 以下.

但是，你是否知道怎样计算 π 的近似值？是否尝试过利用所知道的数学知识来计算 π 的近似值，自己当一回阿基米德或祖冲之？我们建议了下面几种方法. 这些方法很简单，一学就会，你不妨在计算机上试一试，体验一下自己得到 π 值时的喜悦. 相信你学会这些方法之后不会满足于只用来计算 π，而会利用它们来做别的事情. 你也可以自己想出一些别的方法来计算 π. 当然，现在有更快更好的方法计算 π，可以计算出 π 的十万位以上的近似值. 但这些方法就不那么简单了，不可能在这里介绍. 你如果有兴趣，可以尝试从别的资料去了解.

2.1　数值积分法

定积分

$$\int_0^1 \frac{4}{1+x^2}\mathrm{d}x = \pi$$

计算出这个积分的数值，也就得到了 π 的值.

一般的，要计算定积分 $S = \int_a^b f(x)\mathrm{d}x$，也就是计算曲线 $y = f(x)$ 与直线 $y = 0, x = a, x = b$ 所围成的曲边梯形 T 的面积. 为此，用一组平行于 y 轴的直线 $x = x_i$（$1 \leqslant i \leqslant n-1$，$a = x_0 < x_1 < x_2 < \cdots < x_{n-1} < x_n = b$）将曲边梯形 T 分成 n 个小曲边梯形，总面积 S 分成这些小曲边梯形的面积之和. 如果取 n 很大，使每个小曲边梯形的宽度都很小，可以将它上方的边界 $f(x)$（$x_{i-1} \leqslant i \leqslant x_i$）近似地看作直线段，将每个小曲边梯形近似地当作梯形来求面积，就得到梯形公式. 如果更准确些，将每个小曲边梯形的上边界近似地看作抛物线段，就得到辛普森公式. 具体公式如下：

梯形公式. 设分点 $x_1,\cdots,x_{n-1}$ 将积分区间 $[a,b]$ n 等分，即 $x_i = a + i(b-a)/n$ ，$0 \leqslant i \leqslant n$. 所有的曲边梯形的宽度都是 $h=(b-a)/n$. 记 $y_i = f(x_i)$. 则第 i 个曲边梯形的面积 S_i 近似地等于梯形面积 $\frac{1}{2}(y_{i-1}+y_i)h$. 将所有这些梯形面积加起来就得到

$$S \approx \frac{b-a}{n}\left[y_1 + y_2 + \cdots + y_{n-1} + \frac{y_0+y_n}{2}\right]$$

这就是梯形公式.

辛普森(Simpson)**公式**. 仍用分点 $x_i = a + i(b-a)/n$ ($1 \leqslant i \leqslant n-1$) 将区间 $[a,b]$ 分成 n 等份，直线 $x = x_i$ ($1 \leqslant i \leqslant n-1$)将曲边梯形分成 n 个小曲边梯形. 再作每个小区间 $[x_{i-1}, x_i]$ 的中点 $x_{i-\frac{1}{2}} = a + \left(i-\frac{1}{2}\right)(b-a)/n$. 将第 i 个小曲边梯形的上边界 $y=f(x)$ ($x_{i-1} \leqslant x \leqslant x_i$) 近似地看作经过三点 $(x, f(x))$ ($x = x_{i-1}, x_{i-\frac{1}{2}}, x_i$)的抛物线段，则可求得

$$S_i \approx \frac{b-a}{n}(y_{i-1} + 4y_{i-\frac{1}{2}} + y_i)$$

其中 $y_{i-\frac{1}{2}} = f(x_{i-\frac{1}{2}})$. 于是得到

$$S \approx \frac{b-a}{6n}[(y_0+y_n) + 2(y_1+y_2+\cdots+y_{n-1}) + 4(y_{\frac{1}{2}} + y_{\frac{3}{2}} + \cdots + y_{n-\frac{1}{2}})]$$

这就是辛普森(Simpson)公式.

练习 1

(1)选取不同的 n ，用梯形公式和辛普森公式计算 $\pi = \int_0^1 \frac{4}{1+x^2}\mathrm{d}x$ 的近似值.

(2)误差的实际观察：选取 $n = 1\,000, 10\,000, 100\,000, \cdots$ 等，观察 n 值的增加所导致的 S 值的变化情况，直到 n 的增加所导致的 S 的变化小于给定的误差界. 比较同一个 n 值下梯形公式和辛普森公式计算结果的差别，对两个方法的精度差别获得一个感性认识.

(3)误差的理论分析：参考教科书或参考资料，从理论上分析计算误差与 n 的关系，与实际观察结果相比较.

(4)半径为 1 的圆的面积等于 π . 用数值积分法计算这个面积.

(5)自己设计类似的方法计算 $\lg 2$ 的近似值.

2.2 泰勒(Taylor)级数法

利用反正切函数的泰勒级数

$$\arctan x = x - \frac{x^3}{3} + \frac{x^5}{5} - \cdots + (-1)^{k-1}\frac{x^{2k-1}}{2k-1} + \cdots$$

计算 π.

将 $x = 1$ 代入上面的级数可以得到

$$\frac{\pi}{4} = \arctan 1 = 1 - \frac{1}{3} + \frac{1}{5} - \cdots$$

这似乎可以用来计算 π. 但是,这个无穷序列收敛太慢,不实用. 要使泰勒级数收敛快, $|x|$ 应当比 1 小,最好是远比 1 小. 比如, $\arctan \frac{1}{5}$ 就收敛得较快. 但算出它的值对计算 π 有何帮助呢? 也就是说: $\alpha = \arctan \frac{1}{5}$ 与 π 有什么关系呢? 我们有 $\tan \alpha = \frac{1}{5}$. 由正切的倍角公式可得 $\tan 2\alpha = \frac{5}{12}$, $\tan 4\alpha = \frac{120}{119}$.

$\tan 4\alpha \approx 1$, $4\alpha \approx \frac{\pi}{4}$. 可以用 4α 作 $\frac{\pi}{4}$ 的近似值. 但这还不够准确. 我们把误差 $\beta = 4\alpha - \frac{\pi}{4}$ 求出来对 4α 进行修正. 由 $\tan 4\alpha = \frac{120}{119}$ 及 $\tan \frac{\pi}{4}$ 可求得

$$\tan \beta = \tan\left(4\alpha - \frac{\pi}{4}\right) = \frac{1}{239}, \quad \beta = \arctan \frac{1}{239}$$

$$\frac{\pi}{4} = 4\alpha - \beta = 4\arctan \frac{1}{5} - \arctan \frac{1}{239}$$

于是得到

$$\pi = 16\arctan \frac{1}{5} - 4\arctan \frac{1}{239}$$

这称为 Maqin 公式. 利用 $\arctan x$ 的泰勒展开式求出 $\arctan \frac{1}{5}$, $\arctan \frac{1}{239}$ 的近似值,就可以由 Maqin 公式求出 π 的近似值了.

泰勒级数是无穷级数,实际计算时必然只能取它的前 n 项,导致截断误差

$$E_n = \left|\arctan x - \left(x - \frac{x^3}{3} + \cdots + (-1)^{n-1}\frac{x^{2n-1}}{2n-1}\right)\right|$$

由于是交错级数,当 $|x| < 1$ 时,可以简单地用 $E_n < \frac{x^{2n+1}}{2n+1}$ 来估计截断误差.

练习 2

(1)利用 Maqin 公式,计算 π 的近似值到给定的精度.与 π 的准确值比较.并与数值积分法得到的结果比较.

如果要计算 π 的前 15 位数字,计算 arctan(1/5) 和 arctan(1/239) 应当取到幂级数展开式的多少项?取几千或几万项是否就能得到高精度的 π 值?

(2)自己设计类似的方法计算 lg 2 .

2.3 蒙特卡罗(Monte Carlo)法

单位圆的面积等于 π. 可以用数值积分公式来计算这个面积的近似值.另一个方法是用蒙特卡罗法,即用随机投点的方法来求这个面积 π 的近似值.具体方法如下:

在平面直角坐标系中,以 $O(0,0)$,$A(1,0)$,$C(1,1)$,$B(0,1)$ 为四个顶点作一个正方形,其面积 $S=1$. 以原点 O 为圆心的单位圆在这个正方形内的部分是圆心角为直角的扇形,面积为 $S_1=\pi/4$. 在这个正方形内随机地投入 n 个点,设其中有 m 个点落在单位扇形内.则

$$\frac{m}{n}\approx\frac{S_1}{S}=\frac{\pi}{4},\quad \pi\approx\frac{4m}{n}$$

随机投点可以这样来实现:任意产生区间$[0,1]$内的一组随机数 x,y ,则 (x,y) 就代表一个随机点 P 的坐标.这个点落在单位扇形内的充分必要条件是 $x^2+y^2\leqslant 1$.

练习 3

(1)取不同的 n 做上面的实验.将所得的 π 的近似值记录下来,与已知的 π 的值比较.

(2)观察 n 的大小对所得结果的精度的影响.可以看到:n 太小,精度太差.但如果 n 太大,从计算机上所得的不是真正的随机数,效果仍不理想.总的说来,这个方法的精度是差的.

另一种用蒙特卡罗法来计算 π 的方法是 1777 年法国数学家蒲丰(Buffon)提出的随机掷针实验.其步骤如下:

(1)取一张白纸,在上面画许多间距为 d 的等距平行线.

(2)取一根长度为 l ($l<d$)的均匀直针,随机地向画有平行线的纸上掷

去，一共投掷 n 次（n 是一个很大的整数）. 观察针和直线相交的次数 m .

(3)由分析知道针和直线相交的概率 $p = 2l/\pi d$. 取 m/n 为 p 的近似值，则

$$\pi \approx \frac{2nl}{md}$$

特别取针的长度 $l = d/2$ 时，$\pi \approx n/m$.

练习 4

真正去做大量掷针的实验是很费时间的. 请尝试设计一个方案，用计算机模拟布丰掷针实验，得出 π 的近似值.

练习 5

随机整数互素的概率：

取一个大的整数 N . 在 1 到 N 之间随机地取一对整数 a,b ，找出它们的最大公约数 (a,b) . 当 $(a,b) = 1$ 时称 a,b 互素. 做 n 次这样的实验，记录其中 $(a,b) = 1$ 的情况出现的次数 m . 算出 $p = m/n$ 的值.

理论分析指出：随机整数互素的概率

$$p = \frac{1}{\frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots} = \frac{6}{\pi^2}, \qquad \text{因而} \quad \pi \approx \sqrt{\frac{6n}{m}}$$

练习 6

在三维空间中，由三个圆柱面 $x^2 + y^2 = 1, x^2 + z^2 = 1, y^2 + z^2 = 1$ 围成一个立体. 利用蒙特卡罗法求它的体积.

附录　Mathematica 程序

1. 数值积分法计算 π

```
a = 0;b = 1;y[x_] := 4/(1 + x^2);
n = 1000;
pis1 = N[(b - a)/n * (Sum[y[a + i * (b - a)/n],{i,1,n - 1}]
```

```
                    + (y[a] + y[b])/2),50]
pis2 = N[(b - a)/6/
        n * ((y[a] + y[b]) + 2 * Sum[y[a + i * (b - a)/n],
                                      {i,1,n - 1}]
           + 4 * Sum[y[a + (i - 1/2) * (b - a)/n],
                       {i,1,n}]),50]
```

2. Maqin 公式计算 π

```
n = 100;
pimaqin = N[
          16 * Sum[(-1)^(k-1) * (1/5)^(2k-1)/(2k-1),
                   {k,1,n}]
        - 4 * Sum[(-1)^(k-1) * (1/239)^(2k-1)/(2k-1),
                  {k,1,n}],50]
```

3. 蒙特卡罗法计算 π

```
n = 10000;
pimontecarlo1 =
  Block[{i,m = 0},For[i = n,i > 0,i --,
      m = m + If[Random[]^2 + Random[]^2 <= 1,1,0]];
          N[4 * m/n,10]]
```

实验三　最佳分数近似值

人们常称赞祖冲之求出了 π 的近似值 3.141 592，精确度达到小数点后面 6 位，误差在 0.000 001 以下. 但事实是：祖冲之求出的并不是小数近似值，而是分数近似值 355/113. 小数近似值 3.141 592 是将祖冲之的分数近似值 355/113 展开而得到的，以便说明祖冲之的圆周率的精度是多么的高，误差是多么的小. 然而，我们将看到，在某种意义上，展开后得到的这个小数近似值 3.141 592 远不如祖冲之原来的分数近似值 355/113 好.

π 是无理数. 对于任何一个无理数 α，不可能用分数 p/q 来作它的准确值，只能作它的近似值. 近似值 p/q 的好坏可以用绝对误差 $\Delta = |\alpha - p/q|$ 来衡量. Δ 越小，就说明这个近似值 p/q 的精确度越高. 对于给定的分母 q，总可选择适当的分子 p 使 p/q 最接近 α，也就是使误差 Δ 最小. 此时一定有 $\Delta < 1/2q$. 由此可见，要提高精确度，减少误差，一个简单的办法就是增大分母 q. 只要 q 足够大，就可以使误差任意小. 但能不能用较小的分母实现较高的精确度呢?好比买东西，一般来说，花越多的钱就能买到越好的东西. 但我们也希望价廉物美，花较少的钱也能买到好东西. 看看祖冲之的圆周率 355/113. 它的分母只有 113，这样大的分母可以保证误差在 $1/226 \approx 0.004\ 4$ 以下，但它作为 π 的近似值的误差 $|\pi - 355/113|$ 却在 0.000 000 3 以下，大大超额完成任务！而它展开得到的小数近似值 3.141 592（不足近似值）或 3.141 593（过剩近似值），误差与 355/113 基本相同，但写成分数形式 3 141 592/1 000 000 或 3 141 593/1 000 000 后的分母 1 000 000 却大得多，是 113 的 8 850 倍. 这好比花了 8 850 倍的价钱去买同样质量的东西，未免太不划算了吧？

祖冲之算出的圆周率有两个，一个是 22/7，称为"约率"；另一个是 355/113，称为"密率". 不但密率是分母小而且误差小的优秀近似值，约率以很小的分母 7 实现了误差 $< 0.001\ 3$ 的精度，仍不失为好的近似值.

本实验是要研究怎样用分数近似值去对给定的无理数作最佳逼近. "最佳"就是既要分母小，又要精确度高. 我们首先对"最佳"给出一个具体而明确的标准. 然后给出一个求最佳近似值的方法.

3.1 分数对无理数的最佳逼近

最佳逼近. 设 α 是无理数，P/Q 是分数. 如果 P/Q 作为 α 的近似值的误差比起分母不超过 Q 的所有其余的分数都小，也就是 $|\alpha - P/Q| < |\alpha - p/q|$ 对所有 $1 \leqslant q \leqslant Q$ 成立，就称 P/Q 给出了 α 的**最佳逼近.**

我们也可以对分母小、精度高提出另外的标准. 比如说，计算绝对误差 $\Delta = |\alpha - p/q|$ 与分母 q 的乘积 $e = q\Delta = |q\alpha - p|$，$e$ 值越小越优. 或者，再进一步强化"分母小"这一要求，用 $E = q^2\Delta = q|q\alpha - p|$ 作衡量标准，E 值越小越优.

练习 1

已知 $\pi = 3.141\ 592\ 653\ 579\cdots$.

(1) 让分母 q 依次取遍 1 到 100 000 的自然数. 对每个分母 q，取 $p = [q\pi + 0.5]$ 作为分子得到一个最接近 π 的分数 p/q.（这里，符号 $[q\pi + 0.5]$ 表示不超过 $q\pi + 0.5$ 的最大整数，它也就是由 $q\pi$ 四舍五入得到的整数.）从这10 万个分数中选出那些对 π 作了最佳逼近的分数.

(2) 从 π 的 10 万个分数近似值 p/q（$1 \leqslant q \leqslant 100\ 000, p = [q\pi + 0.5]$）中，选出使 $E = q^2\Delta = q|q\alpha - p|$ 值 < 1 的那些分数. 并按照 E 值越小越好的标准选出"十佳"：最小的 10 个 E 值所对应的 p/q.

注意观察：祖冲之的圆周率 22/7（约率）和 355/113（密率）是否在你的选优活动中脱颖而出？

3.2 乐音的频率比

声音是由振动物体发出的. 振动频率越高，音调越高. 比如，音乐中

$$1(\text{do}),\quad 2(\text{re}),\quad 3(\text{mi}),\quad 4(\text{fa}),\quad 5(\text{sol}),\quad 6(\text{la}),\quad 7(\text{si}),\quad \dot{1}(\text{do})$$

这 8 个音（用简谱表示）一个比一个高，就是说它们的频率一个比一个高.

问题 已经知道高八度的 $\dot{1}$ 的频率是 1 的 2 倍. 那么，2，3，4，5，6，7 的频率分别是 1 的多少倍呢？

从 1 到 $\dot{1}$ 要经过 12 个半度音. 假设所有的音的地位平等，每升高半度，频

率乘以同一个倍数 k . 从 1 到 $\dot{1}$ 经过 12 个半度,频率变成 2 倍. 可见 $k^{12}=2$, $k=2^{1/12}$. 上述 8 个音中,从 3 到 4 升高半度,从 7 到 $\dot{1}$ 升高半度,其余相邻的音都相差两个半度. 故各音的频率(以音 1 的频率为 1 个单位)应依次为:

$$1,\quad 2^{1/6},\quad 2^{1/3},\quad 2^{5/12},\quad 2^{7/12},\quad 2^{3/4},\quad 2^{11/12},\quad 2$$

这样一来,各音的频率比都是无理数. 但另一方面,为了使音乐和谐,各音之间的频率比应当是简单的分数,分母越小越和谐. 怎样解决这个矛盾? 用分数对它们作最佳逼近!

练习 2

按照练习 1 所说方法,用适当的分数对上述各音的频率作最佳逼近,重新确定各音的频率比(以音 1 的频率为 1 个单位)使它们都是分母在 8 以内的分数.

用计算机按前面所说的频率($2^{1/12}$ 的各次幂或者它们的分数近似值)产生乐音并在喇叭中播送出来,或者再进一步将它们编成乐曲,听一听它是否是你预料的效果?

3.3 实数的连分数展开

问题 给定一个正实数 α ,怎样求出它的最佳的分数近似值?

像上面所说的那样去大海捞针,对每一个分母依次审查,"万里挑一"选出最佳逼近 α 的分数近似值,显然是一个太笨的办法. 在让你尝试过笨办法之后,下面再介绍一种巧办法,可以直接找到最佳的分数近似值.

按照通用的符号,用 $[\alpha]$ 表示 α 的整数部分,即不超过 α 的最大整数. $a_0=[\alpha]$ 可以作为 α 的最初的分数近似值,其误差 $\varepsilon_1=\alpha-a_0$ 满足条件 $0\leqslant\varepsilon_1<1$. 如果 $\varepsilon_1=0$,则 a_0 就是 α 的准确值. 否则 $0<\varepsilon_1<1$,取 $A_1=1/\varepsilon_1>1$, 则 $\varepsilon_1=1/A_1$,

$$\alpha=a_0+\varepsilon_1=a_0+\frac{1}{A_1}$$

问题转化成寻找 A_1 的分数近似值. 将刚才对 α 做的事情对 A_1 重复一遍: 取 $a_1=[A_1]$, $0\leqslant\varepsilon_2=A_1-a_1<1$. 取 a_1 作为 A_1 的近似值,则

$$\alpha_1=a_0+\frac{1}{a_1}$$

是 α 的分数近似值. 如果 $\varepsilon_2=0$,则 $\alpha=\alpha_1$. 否则再取 $A_2=1/\varepsilon_2>1$,

$$A_1 = a_1 + \frac{1}{A_2}$$

这一过程可以按下面的递推法则进行下去：

令 $A_0 = \alpha$.

对每个整数 $n \geqslant 0$，如果已经求出非负整数 a_k（$0 \leqslant k \leqslant n-1$）及非负实数 A_k（$0 \leqslant k \leqslant n$）使 $a_k = [A_k] < A_k$，$0 < \varepsilon_{k+1} = A_k - a_k < 1$ 和 $A_{k+1} = 1/\varepsilon_{k-1}$ 对 $0 \leqslant k \leqslant n-1$ 成立. 则取

$$a_n = [A_n], \quad 0 \leqslant \varepsilon_{n+1} = A_n - a_n < 1$$

如果 $\varepsilon_{n+1} = 0$，则递推过程终止.

否则 $\varepsilon_{k-1} > 0$，取

$$A_{k-1} = \frac{1}{\varepsilon_{k-1}} > 1$$

则

$$A_k = a_k + \frac{1}{A_{k+1}}$$

设上述递推过程至少可进行 n 步，得到非负整数列 a_k（$0 \leqslant k \leqslant n-1$）及非负实数列 A_k（$0 \leqslant k \leqslant n$）. 对每个 $0 \leqslant k \leqslant n-1$，将 $k-1$ 个等式 $A_i = a_i + \dfrac{1}{A_{i+1}}$（$1 \leqslant i \leqslant k-1$）依次代入 $\alpha = a_0 + \dfrac{1}{A_1}$，得

$$\alpha = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cdots \cdots \cdots + \cfrac{1}{a_{k-1} + \cfrac{1}{A_k}}}}$$

为了书写的方便，我们将 α 的上述表达式简写为

$$a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_{k-1}} + \frac{1}{A_k}$$

的形式.

可用 a_k 作为 A_k 的近似值，得到 α 的近似值

$$\alpha_k = a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_{k-1}} + \frac{1}{a_k}$$

α_k 的这个表达式称为**有限连分数**，它可以化为普通的最简分数 p_k/q_k，称为 α 的第 $k+1$ 个**渐近分数**. 而 $p_0/q_0 = a_0/1$ 是第 1 个渐近分数.

如果 $\varepsilon_{n+1} = 0$ 对某个 n 成立，即 A_n 等于整数 a_n，则有限连分数

$$\alpha_n = a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_{n-1}} + \frac{1}{a_n}$$

是 α 的准确值，α 是有理数.

如果 α 是无理数，则前述递推过程可以无穷地进行下去，得到无穷的整数列 $a_0,a_1,a_2,\cdots,a_n,\cdots$. 而 α 被表达为**无限连分数**

$$\alpha = a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_n} + \cdots$$

$$= \lim_{n\to\infty}\left(a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_n}\right)$$

每个渐近分数

$$\alpha_k = a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_k} = \frac{p_k}{q_k}$$

都是 α 的分数近似值.

上面所说的求 α 的连分数展开式的递推方法适宜于用计算机进行. 如果用手工进行，则递推过程中由 ε_k 计算 $A_k = 1/\varepsilon_k$ 这一步的计算量太大. 这一步也就是用 1 除以 ε_k 求商 A_k，但我们的真正目的并不是求商 A_k，而只是求它的整数部分 a_k. 而对手工笔算来说，求 a_k 比求 A_k 容易得多. a_k 就是 1 除以 A_k 作带余除法的商. 带余除法的定义如下：

带余除法 设 α 是实数，β 是正实数，q 是使 $q\beta \leqslant \alpha$ 的最大整数，$0 \leqslant r = \alpha - q\beta < \beta$. 则称用 β 对 α 作带余除法得到商 q 和余数 r.

（注：两个整数相除的带余除法是众所周知的，这里只是把它推广到两个实数相除而已. 请注意，带余除法的商 q 就是准确的商 α/β 的整数部分.）

在求 α 的连分数展开式的时候. 第一步求 $a_0 = \lceil\alpha\rceil$ 及 $\varepsilon_1 = \alpha - a_0$ 很容易. 而且可以认为 a_0,ε_1 分别是 α 除以 1 的整数商和余数. 下一步是求 $A_1 = 1/\varepsilon_1$ 的整数部分 a_1. 我们用 1 除以 ε_1 作带余除法，求出整数商 a_1 和余数 $r_2 = 1 - a_1\varepsilon_1$，$0 \leqslant r_2 < \varepsilon_1$. 则

$$\varepsilon_2 = A_1 - a_1 = \frac{1 - a_1\varepsilon_1}{\varepsilon_1} = \frac{r_2}{\varepsilon_1}$$

当 $\varepsilon_2 > 0$，也就是 $r_2 > 0$ 时，又可得

$$A_2 = \frac{1}{\varepsilon_2} = \frac{\varepsilon_1}{r_2}$$

再将 ε_1 除以 r_2 作带余除法，得到整数商 a_2 和余数 $r_3 = \varepsilon_1 - a_2r_2$，$0 \leqslant r_3 < r_2$. 于是又有 $\varepsilon_3 = A_2 - a_2 = r_3/r_2$. 当 $r_3 > 0$ 时有 $A_3 = r_2/r_3$. 这个过程可按照下面的递推法则进行下去：

辗转相除法 约定 $r_{-1} = \alpha, r_0 = 1$. 对每个非负整数 k，设已经得到了 $r_i(-1 \leqslant i \leqslant k)$ 和整数 a_i ($0 \leqslant i \leqslant k-1$). 当 $r_k > 0$ 时，用 r_{k-1} 除以 r_k 作带余除法，得到整数商 a_k 和余数 r_{k+1}，$0 \leqslant r_{k+1} < r_k$；当 $r_k = 0$ 时递推过程终止.

已经求得 $a_0,a_1,\cdots,a_k,\cdots$ 之后，如何求各个渐近分数 p_k/q_k？有以下递推

算法：

约定 $p_{-1}=1, q_{-1}=0, p_0=a_0, q_0=1$.

对每个 $k\geqslant 1$，有递推关系式

$$p_k=p_{k-2}+a_k p_{k-1}$$
$$q_k=q_{k-2}+a_k q_{k-1}$$

练习 3

(1)作 $\pi=3.141\,592\,653\,579\cdots$ 的连分数展开，求出前面几个渐近分数.并通过观察回答下面的问题：

i)这些渐近分数是否是练习 1 中选出的对 π 作最佳逼近的近似值？

ii)这些渐近分数 p_k/q_k 是 π 的不足近似值还是过剩近似值？π 的准确值是否在任意两个相邻两个渐近分数之间？

iii)计算相邻两个渐近分数之差

$$\frac{p_k}{q_k}-\frac{p_{k+1}}{q_{k+1}}=\frac{d}{q_k q_{k+1}}$$

的分子 d，由此得出关于 p_k/q_k 的误差的结论

$$\Delta=\left|\pi-\frac{p_k}{q_k}\right|<\frac{1}{q_k q_{k+1}}<\frac{1}{q_k^2}$$
$$E=q_k^2\Delta<1$$

与练习 2(2)的结果相比较.

(2)利用连分数展开将练习 2 重做一遍.

课外思考题　用理论推导证明你在上述练习 3(1)中观察得出的结论对任意实数 α 的渐近分数都成立.

3.4　计算对数值

对给定的正实数 b,N 且 $b\neq 1$，要求对数值 $\alpha=\log_b N$，也就是求实数 α 使 $b^\alpha=N$. 如果能找到整数 p,q 使 $b^p\approx N^q$，则 $b^{p/q}\approx N$，$\log_b N\approx p/q$. 以 $\lg 2$ (即 $\log_{10}2$)为例：由 $2^{10}=1\,024\approx 1\,000=10^3$ 可得 $\lg 2\approx 3/10=0.3$. 再要提高精确度，就要找出更大的 q 使 2^q 更接近 10 的某个幂 10^p，也就是使 $2^q/10^p$ 更接近 1.

练习 4

让 q 依次取遍 1 到 10 000 的所有的正整数. 对每个 q，按如下的递推法则求出一个正整数 $p = p(q)$ 使实数 $\lambda(q) = 2^q/10^p$ 最接近于 1:

$q = 1$ 时，$p(1) = 0$，$\lambda(1) = 2^1/10^0 = 2$.

设已对 q 求出 $p(q)$ 和 $\lambda(q)$. 计算 $2\lambda(q)$（即 $2^{q+1}/10^p$）. 如果 $2\lambda(q) < \sqrt{10}$，则取 $p(q+1) = p(q)$，$\lambda(q+1) = 2\lambda(q)$. 如果 $2\lambda(q) \geqslant \sqrt{10}$，则取 $p(q+1) = p(q) + 1$，$\lambda(q+1) = 2\lambda(q)/10$.

如果 $\lambda(q)$ 比以前所有的 $\lambda(i)$（$1 \leqslant i \leqslant q-1$）都更接近 1，即 $|\lambda(q) - 1| < |\lambda(i) - 1|$ 对所有的 $1 \leqslant i \leqslant q-1$ 成立，就取 p/q 作为 lg 2 的一个近似值. 这样得出的 p/q 都是最佳逼近 lg 2 的分数近似值，它们可以展开成小数近似值. 试自己估计这些近似值的误差范围.

练习 4 的方法当然是一个笨办法. 试一试这个笨办法，你才能更体会到下面的连分数展开法的快捷.

求对数 $\log_b N$ 的辗转相除法 取 $N_{-1} = N$，$N_0 = b$. 对每个整数 $k \geqslant 0$，如果已经知道所有的 N_i（$-1 \leqslant i \leqslant k$）及 a_i（$0 \leqslant i \leqslant k-1$），则可按下面的方法求出 N_{k+1} 和 a_k：

将 N_{k-1} 连续除以若干个 N_k，直到所得的商 $\lambda = N_{k-1}/N_k^{a_k}$ 满足条件 $1 \leqslant \lambda < N_k$ 为止，则连除的 N_k 的个数就是 a_k，而商 λ 就是 N_{k+1}. 如果某个 $N_{n+1} = 1$，则递推过程终止，

$$\log_b N = a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_n}$$

是有限连分数从而是有理数. 否则，递推过程可以无限地进行下去，

$$\log_b N = a_0 + \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_n} + \cdots$$

被展开成无限连分数.

想一想这个方法为什么是正确的?

注意：a_k，N_k 所满足的条件 $1 \leqslant N_{k+1} = N_{k-1}/N_k^{a_k} < N_k$ 也就是 $0 \leqslant \log_b N_{k+1} = \log_b N_{k-1} - a_k \log_b N_k < \log_b N_k$. 这也就是说：$\log_b N_{k-1}$ 除以 $\log_b N_k$ 得到商 a_k 和余数 $\log_b N_{k+1}$. 因此，我们实际上就是在用 3.3 的辗转相除法求 $\log_b N$ 的连分数展开式.

练习 5

用上述辗转相除法求 lg 2 的前几个渐近分数，展开成小数近似值并估计

它们的误差.将所得的结果与练习 4 的结果作比较.

注 计算对数 $\log_b N$ 的通常的方法是利用 Taylor 展开式

$$\ln(1+x)=x-\frac{x^2}{2}+\frac{x^3}{3}-\cdots+(-1)^{n-1}\frac{x^n}{n}+\cdots$$

分别求出 $\ln N$,$\ln b$,再利用换底公式 $\log_b N=\ln N/\ln b$.当 $N=1+x$ 与 1 相差太远时 Taylor 级数不收敛或收敛得慢.此时可选适当的 $N_1,N_2,\cdots,N_k$ 使每个 $N_1,N_2,\cdots,N_k$ 都很接近 1,且 $N=N_1N_2\cdots N_k$,从而 $\ln N=\ln N_1+\ln N_2+\cdots+\ln N_k$.对 $\ln b$ 也可用同样的办法处理.

3.5 二元一次不定方程的整数解

问题 设 a,b,c 是整数,求二元一次方程 $ax+by=c$ 的整数解.

不妨设 a,b 都不为 0,否则方程很容易解.必要时交换未知数 x,y,可化为 $|a|\geqslant|b|$ 的情形,并可使 $a>0$.

利用 3.3 所介绍的辗转相除法,可以得到余数数列 $a,|b|,r_1,r_2,\cdots$ 和商数列 $a_0,a_1,a_2,\cdots$ 使 r_{i-1} 除以 r_i 的商为 a_i,余数为 r_{i+1}.(约定 $a=r_{-1}$,$|b|=r_0$.)由于 r_i 是逐步减少的正整数,必然有某个 $r_{k+1}=0\neq r_k$,余数数列和商数列终止.最后一个非零的 r_k 就是 a,b 的最大公约数 $d=(a,b)$.而分数 $\frac{a}{|b|}$ 被展开成有限连分数

$$\frac{a}{|b|}=a_0+\frac{1}{a_1}+\frac{1}{a_2}+\cdots+\frac{1}{a_{k-1}}+\frac{1}{a_k}$$

去掉这个连分数的最后一项 $\frac{1}{a_k}$,再将所得的连分数化成普通的既约分数 $\frac{u}{v}$,则 $\frac{u}{v}$ 是 $\frac{a}{|b|}$ 的渐近分数近似值:

$$\frac{u}{v}=a_0+\frac{1}{a_1}+\frac{1}{a_2}+\cdots+\frac{1}{a_{k-2}}+\frac{1}{a_{k-1}}$$

误差

$$\left|\frac{a}{|b|}-\frac{u}{v}\right|=\frac{d}{|b|v}$$

于是 $av-|b|u=\varepsilon d$,$\varepsilon=\pm1$.$a(\varepsilon v)-|b|(\varepsilon u)=d$.如果 c 不被 d 整除,则原方程无整数解.否则,$t=\frac{c}{d}$ 是整数,$a(\varepsilon tv)-|b|(\varepsilon tu)=c$.设 $b=\rho|b|$,$\rho=\pm1$.则

$$x_0=\varepsilon tv,y_0=-\varepsilon\rho tu$$

是方程 $ax+by=c$ 的一组整数解. 方程的通解为

$$x=x_0+\frac{\lambda b}{d},\quad y=y_0-\frac{\lambda a}{d}$$

其中 λ 取遍所有整数.

练习 6

求二元一次方程 $1\ 234\ 567x-7\ 654\ 321y=1$ 的最小正整数解.

附录 连分数的性质

设

$$\alpha=a_0+\frac{1}{a_1}+\frac{1}{a_2}+\cdots+\frac{1}{a_n}+\cdots$$

$$\alpha_k=a_0+\frac{1}{a_1}+\frac{1}{a_2}+\cdots+\frac{1}{a_k}=\frac{p_k}{q_k}\quad(\forall\ 1\leqslant k\leqslant n)$$

并定义 $p_0=a_0,q_0=1$，$\alpha_0=p_0/q_0=a_0/1$. 我们希望研究用渐近分数 $p_k/q_k(k\geqslant 0)$ 逼近 α 时的误差.

1. 设 $0\leqslant k\leqslant n-1$. 当 k 为偶数时，$\alpha_k<\alpha<\alpha_{k+1}$. 当 k 为奇数时，$\alpha_k>\alpha>\alpha_{k+1}$. 总之，$\alpha$ 在 α_k 与 α_{k+1} 之间，从而 $|\alpha-\alpha_k|<|\alpha_k-\alpha_{k+1}|$.

证明. 对每个 $0\leqslant i\leqslant k$，记

$$\beta_i=a_i+\frac{1}{a_{i+1}}+\cdots+\frac{1}{a_{k-1}}+\frac{1}{a_k}$$

则 $\beta_k=a_k$，$\beta_{i-1}=a_{i-1}+\dfrac{1}{\beta_i}$，$\alpha_k=\beta_0$.

由 $A_k>a_k=\beta_k$ 得

$$A_{k-1}=a_{k-1}+\frac{1}{A_k}<a_{k-1}+\frac{1}{\beta_k}=\beta_{k-1}$$

再得
$$A_{k-2}=a_{k-2}+\frac{1}{A_{k-1}}>a_{k-2}+\frac{1}{\beta_{k-1}}=\beta_{k-2}$$

照此下去，i 每增加 1，A_{k-i} 与 β_{k-i} 之间的大小关系就颠倒一次. i 为偶数时 $A_{k-i}>\beta_{k-i}$，i 为奇数时则 $A_{k-i}<\beta_{k-i}$. 取 $i=k$ 即可得出 α 与 α_k 之间的大小关系.

2. 取 $p_{-1}=1,q_{-1}=0,p_0=a_0,q_0=1$. 则递推关系式

$$p_k=a_kp_{k-1}+p_{k-2}\quad 和\quad q_k=a_kq_{k-1}+q_{k-2}$$

对 $k\geqslant 1$ 成立.

证明. 对 k 用数学归纳法.

$k=1$ 时,

$$\frac{p_1}{q_1}=a_0+\frac{1}{a_1}=\frac{a_1a_0+1}{a_0}$$

$$p_1=a_1a_0+1=a_1p_0+p_{-1}$$

$$q_1=a_0=a_1q_0+q_{-1}$$

设递推公式对 $1,2,\cdots,k-1$ 已成立. 则在公式

$$\alpha_{k-1}=\frac{p_{k-1}}{q_{k-1}}=\frac{a_{k-1}p_{k-2}+p_{k-3}}{a_{k-1}q_{k-2}+q_{k-3}}$$

中将 a_{k-1} 换成 $a_{k-1}+\dfrac{1}{a_k}$ 就得到

$$\alpha_k=\frac{\left(a_{k-1}+\dfrac{1}{a_k}\right)p_{k-2}+p_{k-3}}{\left(a_{k-1}+\dfrac{1}{a_k}\right)q_{k-2}+q_{k-3}}$$

$$=\frac{p_{k-1}+\dfrac{1}{a_k}p_{k-2}}{q_{k-1}+\dfrac{1}{a_k}q_{k-2}}$$

$$=\frac{a_kp_{k-1}+p_{k-2}}{a_kq_{k-1}+q_{k-2}}$$

于是

$$p_k=a_kp_{k-1}+p_{k-2}\quad 和\quad q_k=a_kq_{k-1}+q_{k-2}$$

成立.

3. 对 $0\leqslant k\leqslant n-1$, $p_k/q_k-p_{k+1}/q_{k+1}=d_k/(q_kq_{k-1})$ 的分子

$$d_k=p_kq_{k-1}-q_kp_{k+1}=(-1)^{k-1}$$

从而

$$\left|\alpha-\frac{p_k}{q_k}\right|<\frac{1}{q_kq_{k+1}}<\frac{1}{q_k^2}$$

证明. 对 k 用数学归纳法.

当 $k=0$ 时, $d_0=p_0q_1-q_0p_1=a_0a_1-1(a_1a_0+1)=-1$.

设 $d_i=(-1)^{i+1}$ 对 $0\leqslant i\leqslant k-1$ 已成立. 则

$$\begin{aligned}d_k&=p_kq_{k+1}-q_kp_{k+1}\\&=p_k(a_{k+1}q_k+q_{k-1})-q_k(a_{k+1}p_k+p_{k-1})\\&=p_kq_{k-1}-q_kp_{k-1}\\&=-d_{k-1}=-(-1)^k=(-1)^{k+1}\end{aligned}$$

实验四　数列与极数

极限是微积分乃至整个数学分析最重要的基本内容之一. 远在公元前三世纪，古希腊人Archimedes就采用了数列极限的思想来计算曲边三角形的面积. 本实验的目的是通过计算机发现数列的规律与极限状态的性质.

所谓一个无穷数列是指按一定顺序排列的一串数字

$$a_1, a_2, \cdots, a_n, \cdots \tag{1}$$

而一个无穷级数则是用无穷项数字构成的和式

$$\sum_{n=1}^{\infty} a_n = a_1 + a_2 + \cdots \tag{2}$$

数列与级数有密不可分的关系. 给定一个无穷级数(2)，它唯一地确定了一个无穷数列

$$S_1, S_2, \cdots$$

其中 $S_n = a_1 + \cdots + a_n$, $n = 1, 2, \cdots$. 反过来，给定一个无穷数列(1)，它也唯一地确定了一个无穷级数

$$\sum_{n=1}^{\infty} b_n$$

这里 $b_1 = a_1$, $b_n = a_n - a_{n-1}$, $n = 2, 3, \cdots$. 并且，无穷级数的和就是相应的无穷数列的极限。因此，无穷数列与无穷级数是可以相互转化的.

给定的数列 $\{a_n\}$ ，人们最关心的问题是：

1. 数列 a_n 有什么规律与性质?

2. 当 $n \to \infty$ 时，数列 a_n 的极限是什么?

3. 极限是否是一个有限的数字? 还是无穷大? 抑或极本不存在?

4. 如果极限是无穷大，那么它趋于无穷大的阶是多少?

5. 如果数列的极限极本不存在，那它在无穷大时的极限状态又怎么样?

对于给定的一个无穷级数，也可以提出上述类似的问题.

本实验将通过计算机图示的方法来帮助我们发现数列的规律及其极限行为. 我们以 Fibonacci 数列为例来探讨上述问题.

4.1　Fibonacci 数列

给定如下的数列

$$1,1,2,3,5,8,13,21,34,55,89,\cdots$$

其递推关系式由

$$F_{n+2}=F_{n+1}+F_n,\quad n=1,2,\cdots,\quad F_1=1,F_2=1 \tag{3}$$

给出. 该数列被称为 Fibonacci 数列.

Fibonacci 数列经常以著名的养兔问题提出来. 某人养了一对兔子(公母各一只). 一月后,这对兔子生了一对小兔. 以后每月,每对成熟(即一月以上)的兔子都生育一对小兔. 假设兔子不会死亡,问一年后总共有多少对兔子? 显然,问题的答案就是数列的第十二项.

为考察 Fibonacci 数列的极限与规律,我们用计算机计算出 Fibonacci 数列每一项的值,并在二维平面上画出顺次连接点 (n,F_n) , $n=1,2,\cdots,N$ 的折线图,其中 N 是一个大整数.

练习 1

分别取 $N=20,50,100,200,500$,观察 Fibonacci 数列的折线图. Fibonacci数列是否单调增? 它是否趋于无穷? 它增加的速度是快还是慢?你能否证实你的观察?

为进一步研究 Fibonacci 数列 F_n 的特性,我们将 F_n 取对数,在直角坐标系中画出顺次连接点 $(n,\log(F_n))$, $n=1,2,\cdots,N$ 的折线图. 此时的折线图近乎于一条直线. 因此,我们猜测 $\log(F_n)$ 是 n 的线性函数. 取 $N=1\ 000$,对上述数据进行拟合可得

$$\log(F_n)\approx-0.803\ 903+0.481\ 211\ n \tag{4}$$

故

$$F_n\approx 0.447\ 567\cdot 1.618\ 03^n \tag{5}$$

练习 2

分别取 $N=2\ 000,\ 5\ 000,\ 10\ 000$,用直线去拟合数据 $(n,\log(F_n))$, $n=1,2,\cdots,N$,由此求数列 F_n 的近似表示. 注意观察 $\log(F_n)$ 的线性项的系数,它与黄金分割数有何联系?

由计算机观察到的上述结果我们似乎可以猜测数列 F_n 的通项具有形式

$$F_n=cr^n \tag{6}$$

将上式代入递推公式(3)得

$$r^2 = r + 1 \tag{7}$$

从而 $r = \frac{1 \pm \sqrt{5}}{2}$. 因为数列趋于无穷，故取 $r = \frac{1 + \sqrt{5}}{2}$. 于是

$$F_n = c\left(\frac{1 + \sqrt{5}}{2}\right)^n \tag{8}$$

然而，公式(8)并不满足 $F_1 = F_2 = 1$，即并非数列 F_n 的通项公式. 不过，它仍然是数列 F_n 的主项.

练习 3

证明公式(8)不是 Fibonacci 数列的通项.

为进一步得到 Fibonacci 数列的通项，我们构造数列

$$b_n = F_n - cr^n$$

将上式代入公式(3)可得数列 b_n 仍然满足递推公式(3). 因而我们猜测，数列 b_n 的通项也具有形式

$$b_n = \bar{c}\,\bar{r}^n$$

其中 $\bar{r}$ 也满足方程(7)，故 $\bar{r} = \frac{1 - \sqrt{5}}{2}$. 这样，我们得到 Fibonacci 数列的通项的一个新的猜测

$$F_n = c\,r^n + \bar{c}\,\bar{r}^n$$

由条件 $F_1 = F_2 = 1$ 确定出 $c = \frac{1}{\sqrt{5}}$, $\bar{c} = -\frac{1}{\sqrt{5}}$，从而我们得到

$$F_n = \frac{1}{\sqrt{5}}\left(\left(\frac{1 + \sqrt{5}}{2}\right)^n - \left(\frac{1 - \sqrt{5}}{2}\right)^n\right) \tag{9}$$

这样，Fibonacci 数列趋于无穷的阶为 $\frac{1}{\sqrt{5}}\left(\frac{\sqrt{5} + 1}{2}\right)^n$.

练习 4

验证(9)式正是 Fibonacci 数列的通项公式.

Fibonacci 数列与自然界中的许多现象，如植物的枝干与叶子的生长有紧密的联系. 它在纯数学领域的一个极为成功的应用是协助前苏联数学家马蒂雅舍维奇解决了著名的 Hilbert 第十问题. 此外，它在优化、运筹以及计算机科学与艺术领域都具有极大的应用价值. 下面我们来“听一听”Fibonacci 数列.

练习 5

取一整数 m (如 $m=51$),将 Fibonacci 数列模 m 得到一周期数列,将该周期数列的值作为音高,编程演奏它.取不同的 m,或将几段合并,感受旋律的变化.

4.2 调和级数

熟知,无穷级数

$$\sum_{n=1}^{\infty} \frac{1}{n^{\alpha}} \tag{10}$$

当 $\alpha>1$ 时收敛,当 $\alpha \leqslant 1$ 时发散。特别地,当 $\alpha=1$ 时,级数(10)称为**调和级数**.

一个令人感兴趣的问题是,调和级数发散到无穷的速度有多快?或者说数列

$$S_n = 1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$$

趋于无穷的速度有多快?

一个直观的方法仍然是画出由点 (n,S_n) ,$n=1,2,\cdots,N$ 构成的折线图.

练习 6

取充分大的 N,观察调和级数的折线图.你觉得它发散的速度是快还是慢?将它的图形与 $y=x$,$y=\sqrt{x}$ 以及 $y=\sqrt[4]{x}$ 做比较,谁的发散速度快?

从上述实验的结果看出,调和级数发散的速度较慢.但是,它到底以什么样的速度发散到无穷?让我们再做下面的练习.

练习 7

对充分大的一系列 n,计算 $S_{2n}-S_n$,你能否猜测出 $S_{2n}-S_n$ 当 n 趋于无穷的极限?更一般地,$S_{2^k n}-S_n$ 趋于无穷的极限是什么?反过来,固定 n,让 k 趋于无穷,$S_{2^k n}$ 趋于无穷的速度是什么?你能否由此得出 S_n 当 n 趋于无穷的极限阶?

我们也可以从另一个角度考察上述问题.

练习 8

用 $J(n)$ 表示不小于 S_n 的最小整数.

1. 对 $n = 1, 2, \cdots, N$（N 是某个大整数），计算 $J(2n) - J(n)$. 你能做出什么猜测？对每个 n，设 $J(m) = J(n) + 1$，则 m/n 的范围是什么？

2. 对每个 $m = 1, 2, \cdots, 30$，令 n 是使得 $J(n) = m$ 成立的最大整数，我们把它记为 $L(m)$. 试计算比值 $L(m+1)/L(m)$. 你能据此做出何种猜测？当 m 趋于无穷时，$L(m)$ 关于 m 的阶是多大？由此，S_n 关于 n 的极限阶是多少？

对调和级数做更仔细的分析，可以得到更精细的结果. 有兴趣的读者不妨做进一步的探讨.

4.3 思考问题

作为本实验的实验内容，请读者研究下列数列的极限状态与规律.

问题 1 设 $a_{n+1} = a_n + \dfrac{1}{a_n}$，$a_1 = 1$，研究数列 a_n 的极限行为.

(1)在平面上画出顺次连接点 (n, a_n)，$n = 1, 2, \cdots, 2\ 000$ 的折线图.

(2)根据上述图形，你认为数列 a_n 的极限是什么？

(3)用一恰当的函数 $y = f(x)$ 去拟合上述图形.

(4)猜测数列 a_n 的极限阶.

(5)你能否证明你的结论？

问题 2 研究数列 $a_n = \sin(n)$ 的极限状态的规律.

(1)在平面上画出点列 (n, a_n)，$n = 1, 2, \cdots, N$（如 $N = 5\ 000$）.

(2)根据上述图形，你认为数列 a_n 的极限是否存在？

(3)你能从上述图形中观察到点列的分布有什么规律？

(4)你能否证明所观察到的规律？

(5)任取区间 $[a, b] \subset [-1, 1]$，画出数列中落在区间 $[a, b]$ 中的点. 将区间 $[a, b]$ 放大并取不同的 N，观察落在区间 $[a, b]$ 中的点集有何变化.

(6)根据以上观察，你认为数列 a_n 的聚点集合是什么？你能否证明你的结论？

问题 3 考察由如下关系确定的正整数数列

$$x_{n+1}=\begin{cases}3x_n+1 & \text{如果 } x_n \text{ 为奇数}\\ \dfrac{x_n}{2} & \text{如果 } x_n \text{ 为偶数}\end{cases}$$

任取一正整数作为初值 x_0，计算数列 x_n，并在平面坐标系中用折线连接 (n,x_n)，$n=0,1,\cdots,N$．取不同的初值 x_0，观察所得的结果．你能发现数列 x_n 有什么规律吗？你能否尝试证明你所发现的规律？

问题 4 研究 Farey 数列的规律与项数．

给定整数 n，将分母不超过 n 的所有真分数（以最简分数形式出现）从小到大排列，所得到的数列称为 n 阶 Farey 数列．例如，6 阶 Farey 数列是 $\frac{1}{6}$，$\frac{1}{5}$，$\frac{1}{4}$，$\frac{1}{3}$，$\frac{2}{5}$，$\frac{1}{2}$，$\frac{3}{5}$，$\frac{2}{3}$，$\frac{3}{4}$，$\frac{4}{5}$，$\frac{5}{6}$．对 n 阶 Farey 数列，我们要问它有多少项？相邻各项之间有何联系？为研究这些问题，请做以下实验．

(1)任取一 Farey 数列，考察任意相邻三项之间的关系．将头尾两项的分子分母分别相加，所得分数是什么？你能因此做出什么样的猜想？你能否证明你的猜想？

(2)考察 Farey 数列相邻两项的差，你能得到什么结论？你能否证明你的结论？

(3)将 Farey 数列的每一项减去 $\frac{1}{2}$，所得到的新数列有什么性质？

(4)用 a_n 表 n 阶 Farey 数列 F_n 的项数．观察 n 阶 Farey 数列和 $n+1$ 阶 Farey 数列的关系．由此，a_n 与 a_{n-1} 有何关系？

(5)将点列 (n,a_n)，$n=2,3,\cdots,N$（如 $N=1\,000$）标在二维坐标平面上（可用折线将它们连接起来）．猜测 a_n 与 n 的函数关系，并拟合之．a_n 的极限阶是多少？注意，极限阶前的常数与 $\frac{1}{\pi^2}$ 有何关系？

(6)用 $F_{n,k}$ 表示 F_n 的第 k 项，并令 $D_n=\sum_{k=1}^{a_n-1}\left|F_{n,k}-\frac{k}{a_n}\right|$．用 (5) 类似的办法估计 D_n 的极限阶．据此，你能做出何种猜测？在这里，你要非常小心．你做出的猜测很可能与著名的 Riemann 猜想（见有关素数的实验）有某种联系．

问题 5 在调和级数中，将分母的十进制表示中含有数字 9 的项去掉，由此得到的级数是收敛还是发散呢？请根据本实验中介绍的方法做仔细的分析．

问题 6 考察 $\alpha=2$ 时，级数(10)的一些结果．

(1)对充分大的 N，计算级数的前 N 项和，并计算它与 π^2 的比值．你能否据此猜测级数(10)的和？

(2)设 $p_1=2$，$p_2=3$，$\cdots$ 是按顺序排列的素数．考察无穷乘积

$$\left(1-\frac{1}{p_1^2}\right)\left(1-\frac{1}{p_2^2}\right)\cdots\left(1-\frac{1}{p_n^2}\right)\cdots \tag{11}$$

试计算该无穷乘积的近似值.这个值与(1)中级数的和有何关系?由此,你能做出什么样的猜测?你能否证明你的猜测?

(3)你能否猜测,对一般的 $\alpha > 1$,无穷级数(10)的和与哪个无穷乘积相等?

需要再次提醒读者,本问题与 Riemann 猜想也有着千丝万缕的联系.

附录 Mathematica 程序

下面是本实验中的有关 Mathematica 程序.使用时,只要输入并调用相关函数即可.

1. 画 Fibonacci 数列折线图的函数

```
FibShow[n_Integer] :=
  Module[
  {t = {},i},
  For[i = 1,i <= n,i ++,AppendTo[t,{i,Fibonacci[i]}]];
  ListPlot[t,PlotJoined -> True]
  ]
```

2. 用直线去拟合 $(i,\log(F_i))$, $i = 1,2,\cdots,n$ 的函数

```
FibFit[n_Integer] :=
  Module[
  {t = {},i},
  For[i=1,i<=n,i ++,AppendTo[t,{i,Log[Fibonacci[i]]}]];
  Fit[t,{1,x},x]
  ]
```

3. 演奏 Fibonacci 数列的函数

```
FibPlay[n_Integer]:=
  Module[
  {t = {},i},
  For[i = 1,i <= n,i ++,AppendTo[t,Mod[Fibonacci[i],n]]];
  ListPlay[t,PlayRange -> {0,n},SampleRate -> 5]
  ]
```

4. 显示点列 $(i,\sin(i))$ ，$i = 1,2,\cdots,n$ 的函数

```
PlotList[n_Integer]:=
  Module[
  {t = {},i},
  For[i = 1,i <= n,i ++,AppendTo[t,{i,Sin[i]}]];
  ListPlot[t,PlotStyle -> {PointSize[0.005]}]
  ]
```

5. 级数(10)的部分和

```
HamoSum[n_Integer,m_Integer]:=
Module[{i},
      Sum[1/i^m,{i,1,n}]
      ]
```

6. 级数(11)的部分积

```
PrimeProd[n_Integer]:=
  Module[{i},N[Product[(1 - 1/Prime[i]^2),{i,1,n}]]
  ]
```

参考读物

1. 阿尔伯特. H. 贝勒著. 数论妙趣——数学女王的盛情款待. 谈祥柏译. 上海:上海教育出版社,1997

2. 基斯. 德夫林著. 数学:新的黄金时代. 李文林,袁向东,李家宏,包芳勋等译. 上海:上海教育出版社,1997

实验五 素　　数

伟大的德国数学家高斯说过，数学是科学的女王，而数论则是数学的女王. 在数论这一充满了趣味而布满荆棘的领域中，有关素数的问题(如著名的Goldbach猜想)始终是最富有魅力最吸引人的研究问题. 本实验将探讨素数的规律及其相关的某些有趣问题. 具体地，我们将研究有关素数的下列问题：

- 素数表的构造
- 素数的判别
- 最大的素数
- 构造生成素数的公式
- 素数的分布

我们希望通过本实验激发你对数论的好奇心，使你对自然数的神奇规律而折服，同时使你认识到探索自然数规律的艰难性.

5.1 素数的判别与求解

熟知，如果一个大于 1 的自然数只能被 1 及它本身整除，则该数称为素数. 否则则被称为合数. 从数学史的黎明时期开始，数学家就一直在探索自然数的奥秘. 远在古希腊时代，欧几里得就证明了每一个合数都可以分解为若干个素数的乘积，并且在不计较素数排列顺序时这种分解是唯一的，这就是所谓的**算术基本定理**. 算术基本定理表明，素数是构造自然数的基石，正如物质的基本粒子一样. 正是由于素数如此重要的地位才使得一代又一代数学家努力地探询素数的规律. 首先，一个最基本的问题是

素数到底有多少个?

会不会在某一充分大的自然数以后就没有素数呢?让我们先做以下实验.

练习 1

假设已知前 n 个素数，它们按从小到大的顺序排列为 $p_1 = 2, p_2 = 3, \cdots, p_n$. 对 $n = 1, 2, \cdots, 20$，计算 $N_n = p_1 p_2 \cdots p_n + 1$. 问：1) N_n 是否都是素数? 2)如果 N_n 不是素数，N_n 是否含有不同于 p_i，$i = 1, 2, \cdots, n$ 的素因子?

练习 2

根据练习 1 的结果，猜测素数是否有无穷多个？试给出你的证明.

关于素数的下一个基本问题是

如何求出小于某一给定整数的所有素数？

古希腊另一位学者 Eratosthenes 给出了解决这一问题的方法，这一方法被后人称为 Eratosthenes 筛法.

Eratosthenes 筛法的基本思想是，将自然数列从 2 开始按顺序排列至某一整数 N. 首先，从上述数列中划除所有 2 的倍数（不包括 2）. 在剩下的数中，除 2 外最小的是 3. 接着，从数列中划掉所有 3 的倍数（不包括 3）. 然后在剩下的数中，再划去 5 的倍数……. 这个过程一直进行下去. 则最后剩下的数就是不超过 N 的所有素数. 借用 Eratosthenes 筛法，经过众多学者的艰辛努力，D. N. Lehmer 于 1914 年编织出了 10 000 000 以内的素数表.

练习 3

利用 Eratosthenes 筛法，手工编写 100 以内的素数表. 据此，你如何估量利用筛法编写 10 000 000 以内的素数表的工作量？

练习 4

利用 Eratosthenes 筛法，通过计算机编程求 10 000 以内的所有素数.

筛法是用乘法寻找素数. 实际上，也可以用除法判别一个数是否是素数. 而且，用除法的效率可能会更高. 假设我们已经找到了前 n 个素数 $p_1 = 2$，$p_2 = 3, \cdots, p_n$，为了寻找下一个素数我们从 $p_n + 2$ 开始依次检验每一个整数 N，看 N 是否能被某个 p_i，$i = 1, 2, \cdots, n$ 整除. 如果 N 能被前而的某个素数整除，则 N 为合数. 否则 N 即为下一个素数 p_{n+1}. 实际上，为了提高算法的效率，我们不需用前面的每一个素数去试除 N，而只需用不超过 $\sqrt{N}$ 的素数去除就可以了.

练习 5

利用试除方法,通过计算机编程求 10 000 以内的所有素数. 试将试除法与筛法进行比较,哪一个更有效?

虽然从理论上来说,Eratosthenes 筛法和试除方法可以求出所有的素数,但是通过上面的练习你可能会发现,利用这些方法构造大的素数表是不切实际的. 例如,要构造 10^{50} 以内的素数表,利用当今最快的计算机也得需要一百亿年. 实际上,数学工作者一直在致力扩大素数表的范围. 寻找更大的素数不仅是数学爱好者的乐趣,更是实际应用工作者(如密码编码者)努力追寻的事. 数十年前,象 11…1 (23 个 1)以及 $2^{127}-1$ 的素性判别问题难倒了许多睿智的数学家. 当今,数学家研究出了一套非常复杂而高深的技巧来对数的素性进行检验. 寻找到的最大素数的高峰一股劲地向上无限攀升. 截止到 1996 年 11 月,所找到的最大素数是 $2^{1\,398\,269}-1$,其十进制形式有 40 万位! 为了领略一下隐藏在这些高深技巧下而的数学思想,让我们先做以下实验.

练习 6

对 $n=2,3,\cdots,100$,观察 2^{n-1} 被 n 整除所得的余数. 从观察结果你能得出什么结论?再取其它的整数 m (如 3,4,5),观察 m^{n-1} 被 n 整除的情况. 特别注意观察当 n 为素数时的结果. 你能否因此确信你的结论? 进一步,你所得出的结论的逆命题是否成立?由此,用你的结论能否给出判别一个数是否是素数的判别方法?

练习 7

对互质的整数 $n=2$ 及 $m=1,2,\cdots,1\,000$,求使得 n^d 除 m 的余数为 1 的最小整数 d . 当 m 为素数时,观察 d 与 m 之间的关系,你能得到什么结论? 类似地,对 $n=3,4,5$ 做进一步的观察,你能否确信你的结论? 你所得出的结论的逆命题是否成立?

上述实验表明,给出一个简明的素数判别准则并不容易. 通常需要将更多高深的技巧与之结合才能给出判别素数的更有效方法.

对于具有特殊结构的数的素性判别,有更加快捷的方法. 这方面最引人注目的例子是 Mersenne 数,即形如 2^n-1 的数. 利用 Mersenne 数可以构造出

非常大的素数，如前面指出的最大素数.

练习 8

对 $n=2,3,\cdots,300$，判断哪些 Mersenne 数 $M_n=2^n-1$ 是素数？如果 n 为合数，Mersenne 数 M_n 是素数还是合数？如果 n 为素数，Mersenne 数是否一定是素数？

从练习 8 中可以看到，Mersenne 素数是极其稀少的. 借助大型计算机的威力，截止 96 年 11 月，数学家仅发现了 34 个 Mersenne 素数. 它们对应的 n 是，2，3，5，7，13，17，19，31，61，89，107，127，521，607，1 279，2 203，2 281，3 217，4 253，4 423，9 689，9 941，11 213，19 937，21 701，23 209，44 497，86 243，123 049，216 091，756 839，859 433，1 257 787，1 398 269. 从中可以看出，Mersenne 素数的分布是极不规则的. 如何通过 n 判断 M_n 是否是素数？数学家 Lucas 与 Lehmer 早在 1930 年就给出了具体可行的判别方法. 假定 n 为素数，定义数列

$$u_1=4, u_{k+1}\equiv u_k^2-2 \bmod M_n$$

$k=1,2,\cdots,n$. 如果 $u_{n-1}\equiv 0 \bmod M_n$，则 M_n 为素数. 否则，M_n 为合数. 前面列出的 Mersenne 素数就是在此算法的基础上发现的.

练习9

确定 Mersenne 数 M_7 与 M_{11} 是否是素数.

练习10

有人猜测，如果 n 本身是 Mersenne 素数，则 M_n 也一定是 Mersenne 素数. 你是否相信这一猜测？还有人相信 Mersenne 素数有无穷多个，你的意见呢？

5.2 生成素数的公式

虽然我们在 § 2 给出了生成素数表的算法，但要找到大的素数确是异常的艰辛. 你也许会想，要是能找到一个正好生成全部素数的公式就好了. 例如，是否存在单变量整系数的多项式，它只生成素数并且生成所有的素数？更一般

地，是否存在一个生成素数的多变量函数公式？如果这样的公式不存在，能否找到一个虽不能给出全部但能给出无穷多个素数（且只给出素数）的公式？实际上，早在十七、十八世纪，大数学家 Fermat，Euler 等就研究过这类公式. 1640 年Fermat 在给 Mersenne 的信中指出，对所有的整数 n，$F_n = 2^{2^n} + 1$ 永远是素数. 的确，$F_0 = 3$，$F_1 = 5$，$F_2 = 17$，$F_3 = 257$，$F_4 = 65\ 537$ 都是素数. 然而，好事到此为止. 1732 年，大数学家 Euler 指出，$F_5 = 4\ 294\ 967\ 297$ 不是素数，他并且找到了 F_5 的因子分解. 此后，人们分别证明了 F_6，F_7 与 F_8 都是合数，并得到了它们的素因子分解. 实际上有人猜测 F_n 当 $n > 4$ 时都是合数. 由此可见，Fermat 当初的猜测离真实结果相差有多远.

Fermat 数 F_n 与正多边形做图有紧密的联系. 这一惊人的发现是伟大的德国数学家 Gauss 得到的. Gauss 在 19 岁那年证明了：一个正 n 边行可用直尺与圆规作图的充要条件是，$n = 2^k$ 或者 $n = 2^k p_1 p_2 \cdots p_r$，其中 $p_1, p_2, \cdots, p_r$ 为不同的 Fermat 数. 特别地，正 17 边形可以用直尺与圆规做出.

练习 11

验证 Fermat 数 F_n，$n = 5,6,7,8,9,10$ 均为合数.

1772 年 Euler 研究过用单变量的整系数多项式来生成素数. 其最著名的例子是 $n^2 + n + 41$.

练习 12

对 $n = 0,1,\cdots,100$，计算 $n^2 + n + 41$. 它们是否都给出素数？在 10 000 以内的素数中，由公式 $n^2 + n + 41$ 给出的素数占多少？类似地，对公式 $n^2 - 79n + 1\ 601$ 及 $6n^2 + 6n + 31$ 做同样的判别. 你自己能否给出一个类似的公式？

公式 $n^2 + n + 41$ 有一个非常奇特的性质. 为揭示这一特性，我们考察它的二次求根公式的判别式 $d = 1^2 - 4 \times 1 \times 41 = -163$.　163 有什么特别的地方？有！请看

$$e^{\pi\sqrt{163}} = 262\ 537\ 412\ 640\ 768\ 744.000\ 000\ 000\ 000$$

三个互不相干的无理数产生了一个非常接近整数的数，确实让人吃惊. 你也许会想这不是一个巧合，幕后一定隐藏着什么. 你确是猜对了. 它与 Gauss 关于类数问题的工作紧密相关. 实际上，$d = 163$ 是使 $a + b\sqrt{-d}$（其中 a，b 都是

整数)这样的数系具有唯一因子分解的最大的 d 值.

练习 13

你认为是否存在单变量的整系数多项式 $f(n)$,使得对每个 n,$f(n)$ 都是素数？你能否证明你的结论？

从上面的努力似乎可以看出,利用单变量函数难以得到素数的生成公式.那么,多变量函数是否可行呢？作为 Hilbert 第十问题的一个推论,马蒂雅舍维奇证明了:存在一个多元多项式 $P(x_1,x_2,\cdots,x_n)$,其正值构成的集合恰好是素数的全体.遗憾的是,他并没有给出怎样具体地构造这样的多项式.后经众多数学家的努力,终于在 1977 年构造出了一个具有 26 个变量 25 次的素数生成多项式!

5.3 素数的分布

如果你将素数在数轴标出来,你会发现素数的分布很不规则.让我们通过以下实验做进一步的观察.

练习 14

用 $\pi(n)$ 表示不超过 n 的素数的个数,$\pi(m,n)$ 表示区间 $[m,n]$ 内素数的个数.试计算 $\pi(100)$,$\pi(1\ 000)$,$\pi(10\ 000)$,$\pi(100\ 000)$ 以及 $\pi(100,200)$,$\pi(1\ 000,1100)$,$\pi(10\ 000,10\ 100)$,$\pi(100\ 000,100\ 100)$.从计算结果看,随着整数范围的扩大,素数是越来越稀还是越来越密？选取一些更长的区间,再尝试以上同样的实验.

练习 15

将素数从小到大顺序排列 $p_1=2,p_2=3,\cdots$,用 $d_n=p_{n+1}-p_n$ 表示相邻素数间的间隔.计算 $d_1,d_2,\cdots,d_N$(如 $N=1\ 000,10\ 000$ 等),然后将点 (p_n,d_n) 标在平面坐标系中.你能从中观察到素数的间隔有什么规律吗？譬如,素数的间隔值有哪些？它们各重复多少次？哪些间隔值的重复次数多？最大间隔值是多少？随 N 增大,最大间隔值是否也随之增大？

练习 16

根据上述实验，你对素数的分布能做什么样的猜测？譬如，间隔差为 2 的素数是否有无穷多个？更一般地，间隔差为某个偶数的素数对是否有无穷多个？是否存在相邻的素数，其间隔值可以任意大？你能否证明你的这些猜测？

从上面的实验了解到，素数的分布是极不规则的. 它虽然延数轴分布越来越稀疏，但有时素数之间的间隔又很小. 不过，就总的趋势而言，固定区间长度内的素数的个数是越来越少. 下面，我们进一步研究素数个数的变化规律.

练习 17

在二维坐标平面上标出点列 $(n,\pi(n))$，$n=1,2,\cdots,N$（取不同的 N，如 1 000，10 000 等）. 也可以用折线将点列连接起来. 观察 $\pi(n)$ 趋于无穷的趋势，将它同 $y=x$，$y=\sqrt{x}$ 比较. 你能得出什么结论？类似地观察点列 $(n,\pi(n)/n)$，$(n,\pi(n)/\sqrt{n})$ 及 $(n,\pi(n,\pi(n)/(n/\log(n)))$. 你能据此猜测 $\pi(n)$ 趋于无穷的极限阶吗？

练习 18

令

$$Li(n)=\int_2^n\frac{1}{\log x}\mathrm{d}x$$

$$R(n)=1+\sum_{k=1}^{\infty}\frac{1}{k\zeta(k+1)}\frac{(\log n)^k}{k!}$$

其中

$$\zeta(k)=1+\frac{1}{2^k}+\frac{1}{3^k}+\cdots$$

试对一系列充分大的 n，计算 $\pi(n)$，$n/\log(n)$，$n/(\log(n)-1.083\ 66)$，$Li(n)$ 及 $R(n)$. 其中哪一个公式更接近 $\pi(n)$？

关于素数个数的近似公式首先是 Gauss 于 1792 年给出的，但他当时没能给出证明. 直至 1896 才由 Hadamard 与 Poussin 利用复变函数的理论加以证明. 一个更加精确的近似公式是 1859 年由 Riemann 提出的一个著名的猜想

(即所谓的 Riemann 猜想)所导出的. 不过,Riemann 猜想至今仍未被证明,它无疑是数学上最著名的难题之一.

5.4　进一步的问题

关于素数,存在许许多多富于挑战性的问题,吸引了众多的数学家及业余爱好者. 下面我们介绍几个供有兴趣的读者参阅.

1. Goldbach 猜想

Goldbach 是普鲁士派往俄罗斯的一位公使,彼得堡科学院院士. 1742 年他给大数学家 Euler 的信中提出了两个猜想,即每个不小于 6 的偶数都可以表为两个奇素数之和;每个不小于 9 的奇数可以表为三个奇素数之和. Euler 在随后的复信中写道:任何不小于 6 的偶数都是两个奇素数之和,虽然我不能证明它,但我确信无疑这是完全正确的定理. 这就是著名的 Goldbach 猜想的由来. 两百多年来,无数数学家花费了无数心血都未能解决这一问题. 我国数学家陈景润的工作是迄今为止最好的结果. 他证明了:任何一个充分大的偶数可以表为一个素数与另两个素数乘积之和. 请读者对 10 000 以内的偶数验证 Goldhach 猜想.

2. 大整数的素因子分解

正如判断一个大数的素性一样,将一个大整数分解为素因子的乘积是一件相当艰难的事情,迄今尚无一种通用有效的方法. 目前,最有效的素因子分解算法的运算量大约为 $O(\exp(cL^{1/3}\log(L)^{2/3}))$,其中 L 为要分解的整数 N 的位数. 利用现有大型计算机的能力,能够分解的最大整数不能超过 100 位. 例如,至今尚无人能分解 Fermat 数 F_9. 读者能否给出 F_5 的分解?

1994 年,美国数学家 PeterShor 做出了一项惊人的工作. 他指出,如果使用量子计算机,则因子分解算法的运算量仅为 $O(L^2\log(L)\log\log(L))$.

3. 完全数

所谓**完全数**是指它的所有因子(除去它本身)之和等于该完全数. 例如,6 是一个完全数. 因为 6 的因子为 1,2,3 且 $1+2+3=6$. 下一个完全数是 28. 请读者找出 10 000 以内的所有完全数,并对它做素因子分解. 你能据此猜测完全数的通式吗? 完全数与 Mersenne 素数有何联系? 你能由此找到更多的完全数吗? 是否存在奇完全数? 完全数是否有无穷多个?

除 6 以外，完全数都有一个奇妙的特性，就是每个完全数可以表为几个连续奇数的奇次数次方之和. 如 $28 = 1^3 + 3^3$. 请你对你找出的完全数验证此特性.

4. 孪生素数

所谓孪生素数是指差为 2 的两相邻素数. 孪生素数是否有无穷多个?

5. Bertrand 猜测

当 $n > 3$ 时，n 与 $2n - 2$ 之间至少存在一个素数.

6. 青一色数的素性

由 n 个 1 组成的数 11…1 叫做青一色数. 当 n 为何值时，青一色数是素数? 如果青一色数是合数，如何将它做素因子分解?

附录　Mathematica 程序

以下是本实验中需要采用的 Mathematica 程序.

1. 计算 $N_n = p_1p_2\cdots p_n + 1$，并判断 N_n 是否素数

```
NumP[n_Integer] :=
    Module[{i,Num},
            Num = Product[Prime[i],{i,1,n}] + 1;
            Print[Num];
            Print[PrimeQ[Num]];
            Print[FactorInteger[Num]]
            ]

Do[NumP[n],{n,1,20}]
```

2. 用筛法求所有小于等于 n 的素数

```
Sieve[n_Integer] : =
 Module[
  {t = {},i,temp},
  For[i = 2,i <= n,i ++,AppendTo[t,i]];
  For[i = 1,Prime[i] <= Sqrt[n],i ++,temp = Prime[i];
      t = Select[t,(#1 == temp||Mod[#1,temp]! = 0)&]];
  t]
```

3. 用试除法求所有小于等于 n 的素数

```
DivPrime[n_Integer] : =
    Module[
        {t = {},i,j,temp,divided},
        For[i = 2,i <= n,i ++,
            j = 1;divided = False;
            While[Prime[j] <= Sqrt[i]&&(!divided),
        temp = Prime[j];
              divided = (Mod[i,temp] == 0);j = j + 1];
            If[!divided,AppendTo[t,i]]
        ];
        t]
```

4. PowerMod[a,b,n]表示 a^b 被 n 整除所得的余数

5. Mersenne 数素性的判别

```
Mersenne[n_Integer] : =
  Module[
    {M,i,u = 4},
    If[!PrimeQ[n],False,
      M = 2^n - 1;
      For[i = 1,i < n - 1,i ++,u = Mod[u^2 - 2,M]];
      If[u == 0,True,False]]
    ]
```

6. **PrimePi[n]**给出所有小于等于 n 的素数的个数

7. 练习 18 的程序

```
Compare[n_Integer] :=
  Module[{x,k,t = {}},
    Li = NIntegrate[1/Log[E,x],{x,2,n}];
    R = 1 + NSum[1/k/Zeta[k + 1] * Log[E,n]^ k/Factorial[k],
            {k,2,Infinity}];
    AppendTo[t,PrimePi[n]];
    AppendTo[t,N[n/Log[E,n]]];
    AppendTo[t,N[n/(Log[E,n] - 1.08366)]];
    AppendTo[t,Li];
    AppendTo[t,R];
    Print[t]
    ]

Do[Compare[i],{i,100000,900000,100000}]
```

参考读物

1. 阿尔伯特.H.贝勒著.数论妙趣——数学女王的盛情款待.谈祥柏译.上海:上海教育出版社,1997

2. 基斯.德夫林著.数学:新的黄金时代.李文林,袁向东,李家宏,包芳勋等译.上海:上海教育出版社,1997

3. Mount Holyoke College.数学实验室.白峰杉,蔡大用译.高等教育出版社;施普林格出版社,1998

实验六　概　　率

概率，又称或然率，几率，是反映某种事件发生的可能性大小的一种数量指标. 它介于 0 与 1 之间. 例如在相同条件下独立地多次抛掷一枚硬币，则有接近一半的次数会出现国徽朝上，我们就称在抛掷硬币的试验中，国徽朝上这个事件发生的概率为0.5. 概率的理论本来就是对随机的实验现象的描述. 本实验的主要目的是通过观察和分析实验结果来对概率及随机变量的概念和知识增加了解和加深理解.

6.1　概率的古典定义

我们考察抛掷骰子这样一个试验，如果要你回答在一次抛掷中出现一点这样一个事件的概率为多少，大多数人都会回答大约是六分之一. 这是因为我们找不出理由认为出现一点这个事件比出现其它情况更容易发生或更不容易发生，而在一次掷骰子的试验中共会出现六种结果，我们认为每种结果出现的可能性相同，从而得到出现一点这个事件发生的概率为六分之一.

在“等可能性”概念的基础上，很自然地引进古典概率的定义.

定义 1　假设某一试验满足下面的条件：

(1)它的全部可能结果只有有限个，设此数为 N（如在掷骰子的试验中 $N = 6$，在抛硬币的试验中 $N = 2$ 等).

(2)每个结果等可能出现.

则一个恰好包含 M 个结果的事件 A（如在掷骰子的试验中，A 为出现偶数点的事件，则 $M = 3$）的概率定义为：

$$P(A) = \frac{M}{N}$$

现在我们用一个例子来说明如何利用这个定义来解决问题.

例 1　甲、乙两位棋手棋艺相当. 现他们在一项奖金为1 000元的比赛中相遇，比赛为五局三胜制. 已经进行了三局的比赛，结果为甲二胜一负. 现因故要停止比赛，问应该如何分配这1 000元比赛奖金才算公平？

平均分对甲欠公平，全归甲则对乙欠公平. 合理的分法是按一定的比例分配而甲拿大头. 一种看来合理的分法是按已胜局数分，即甲拿2/3，乙拿1/3. 这

种分法合理吗?

练习 1

在甲已经两胜一负的基础上,在计算机上模拟两位棋手以后的比赛,计算他们应得的奖金.由于两位棋手的棋艺相当,可以假定他们在以下每一局的比赛中胜负的机会各半.Mathematica 中有产生 0 或 1 随机数的函数"**Random[Integer]**".用这个函数可以产生随机数 0 或 1,0 与 1 出现的机会各占一半.可以用随机数 1 表示甲棋手胜,而随机数 0 表示乙胜.(也可以用 $[0,1]$ 中的随机实数来模拟两人的胜负,随机数 >0.5 表示甲胜,否则乙胜.)连续模拟1 000 次(或更多的次数),每次模拟到甲乙两方有一方胜了三局为止,按所说方案分配奖金.1 000 次模拟结束后,计算两棋手每次的平均奖金,就是该棋手应得的奖金.

模拟结果,发现并非甲得 2/3,乙得 1/3.

实际上,因为比赛只需再进行两局,则可分出胜负,结果无非是以下四种情况之一:

甲甲,甲乙,乙甲,乙乙

在这四种情况中,最后甲胜的情况有三种,乙胜的情况只有一种,而每种情况发生的可能性是一样的,所以甲最终得到1 000 元奖金的可能性是3/4,而乙最终得到1 000 元奖金的可能性是1/4.所以合理的分法是,甲得750 元,乙得250 元.

这个例子颇给人以启发,即表面上看来简单自然的东西,经过深入的分析后揭示了其不合理之处.

古典概率的局限性是显然的,它只能用于全部试验结果为有限,且等可能性成立的情形.但在某些情况下,这个概念可以推广到试验结果有无限个的情形,这就是所谓的"几何概率".我们以一个例子来说明.

例 2 在平面上的区域 $[-1,1]\times[-1,1]$ 中随意的选取一点,问"选取的点落在单位圆内"这个事件 A 的概率是多少?

平面上的正方形 $[-1,1]\times[-1,1]$ 中的每一点 (x,y) 都是一个可能的试验结果,而这个正方形就是全部可能的结果之集."在平面上的区域 $[-1,1]\times[-1,1]$ 中随意的选取一点"一语,可以理解为这正方形内任一点都是等可能的.按约定,只有在点 (x,y) 落在单位圆内时,事件 A 才发生.因正方形内包含无限个点,古典概率定义无法使用.于是,我们把"等可能性"这概念按本问题的特点引申一下:正方形内同样的面积有同样的概率.全正方

形的面积为 $2\times 2=4$，而单位圆的面积为 π. 按上述引申了的原则，算出事件 A 的概率 $P(A)=\pi/4$.

这样算出的概率被称为"几何概率"，是因为它是基于几何图形的面积，体积，长度等而算出来的. 就本例而言，重要之点在于将等可能性解释或引申为"等面积，等概率".

练习 2

(1)在计算机上列举出同时抛掷三颗骰子的所有可能结果，比较在一次试验中掷出的点数和为 9 与和为 10 这两个事件何者更容易发生.

(2)利用概率的古典定义计算在抛掷一对骰子的试验中，那一种点数和出现的概率最大？那一种点数和出现的概率最小？

(3)试计算下列两个事件的概率，并比较他们的大小.

i)掷 4 次骰子，至少有一次出现一点.

ii)抛掷一对骰子 24 次，至少有一次出现两个一点.

从这个练习中你能得到一些什么？

(4)试计算实验二的蒲丰(Buffon)随机掷针实验中针与线相交的概率(提示：考虑针的中点与最近邻直线的距离 x 和针与直线的夹角 y).

(5)在单位圆内随意地取一条弦，问"弦长超过该圆内接等边三角形的边长 $\sqrt{3}$"这一事件的概率是多少？

6.2 概率的统计定义

我们仍以抛掷骰于为例，在定义古典概率时，我们假设各面出现的机会是等可能的，这就要假设

(1)骰于的质料绝对均匀.

(2)骰于是绝对的正六而体.

(3)掷骰子时离地面有充分的高度.

但在实际问题中是不可能达到这些要求的. 这时，"出现一点"这个事件 A_1 的概率有多大，已无法仅通过一种理论的考虑来确定. 但我们可以通过试验的方法来得到事件 A_1 的概率：设反复地将骰于抛掷大量的次数，例如 n 次，若在这 n 次抛掷中一点共发生了 m_1 次，则称 m_1/n 是 A_1 这个事件在这 n 次试验(每抛掷一次算作一次试验)中的频率，**概率的统计定义**就是将 m_1/n 作为事件 A_1 的概率 $P(A_1)$ 的估计.

这个概念的直观背景很简单：当一个事件发生的可能性大(小)时，如果在同样条件下反复重复这个实验时，则该事件发生的频繁程度就大(小).

现在要问的问题是：对任何一组试验，频率是否相同?我们说：虽然对任何一组试验，频率不会恒等于同一个数，但我们在数学上可以证明：对几乎任何一组试验，当 n 趋向无穷时，频率 m_1/n 趋向同一个数.

虽然我们不能用概率的统计定义来确切地定出任何一个事件的概率，但我们可以用概率的统计定义对一个事件的概率做出估计，同时也可以利用概率的统计定义对某些理论的正确与否进行检验. 设想根据一定的理论、假定等等算出某事件 A 的概率为 p. 这理论或假设是否与实际相符? 我们并无把握. 于是我们可以诉诸实验，即进行大量重复的试验以观察事件 A 发生的频率 m/n. 以 m/n 与 p 的接近程度来判断有关理论的正确与否. 对这类问题的研究属于假设检验，它是数理统计学的一个重要分支. 有兴趣的读者可以参考有关的书籍.

练习 3

(1)设 p 是区间 $[0,1]$ 内任一实数. 在区间 $[0,1]$ 取随机数 λ，则 $\lambda \leqslant p$ 的概率应等于 p. 取 $n = 100, 1\ 000, 10\ 000$ 个这样的随机数 λ，计算 $\lambda \leqslant p$ 的次数 m，看 n/m 是否接近于 p.

(2)利用概率的统计定义，通过计算机模拟本实验练习 2 中的(1)—(3). 模拟的方法是：产生 $[0,1)$ 中的一个随机数 λ 来模拟掷骰子，以 $[6\lambda] + 1$ 作为骰子的点数，(其中 $[6\lambda]$ 表示 6λ 的整数部分). 或者直接用Mathematica的随机函数**Random[Integer,{1,6}]**来模拟产生 1 到 6 的随机数作为骰子的点数. 分别让计算机按照练习 2 的(1),(2),(3)的要求"掷骰子"并计算所得的结果，进行分析，回答所提的问题.

从模拟中你能得到什么结论?

(3)利用例 2 给出的结果与概率的统计定义，通过计算机模拟(分别取n＝100,200,500,1 000,2 000,5 000，各做100 次)对 π 的值做出估计，通过分析它们的误差，你能得到什么结论?

(4)用计算机进行下面的模拟：

(i)在线段 $[0,1]$ 中随机地取一点(即产生区间 $[0,1]$ 内的一个随机数)，共取 n 次(n 分别取100,500,1 000,5 000).

(ii)将线段 $[0,1]$ 分成 n 个互不相交但长度相等的线段，而后计算各小线段中含有(i)中取出的点的个数.

(iii)计算小线段中含有点数恰好为 k (k 取为 0,1,2,3,4,5)的频率.

分析最后的结果.

注：我们在这里只能介绍概率的这两种定义，关于现代概率的公理化定义已经超出了本书的范围，有兴趣的读者可以参考概率论方面的书籍.

6.3　二项分布与 Poisson 分布

在讲分布的概念之前，我们先介绍一下随机变量的概念. **随机变量**就是"其值随机会而定"的变量. 例如在掷骰子的试验中，如果设掷出的点数为 X，则 X 就是一个随机变量，它可以取1，…，6等 6 个值，具体取什么值，要等掷了骰子后才能知道. 那么，对于一个随机变量 X，它取一个值 x（此值在可能的取值范围内）就对应于某个事件，对于此事件就有一个概率值 $p(x)$，它是 x 的一个函数，我们称它为随机变量 X 的**概率分布**.

考虑如下问题：

· 将一枚硬币抛掷 5 次，恰好等到 2 次国徽朝上的概率是多少？

· 抛掷一颗骰子 9 次，恰好等到 4 个 2 点的概率是多少？

· 一个盒子中装有 2 个红球和 3 个白球，有放回的随机抽取 6 次，恰好有 2 次取到红球的的概率是多少？

略加分析即知，上面的几个问题都是下面这个问题的特例.

· 设某事件 A 在一次试验中发生的概率为 p. 现把这个试验独立地重复 n 次，以 X 记 A 在这 n 次试验中发生的次数，求 X 恰好为 k 的概率.

设我们对 n 次试验结果都做记录，某次试验中如事件 A 发生，则记为 A，如事件 A 不发生，则记为 $\overline{A}$. 要想 X 正好等于 k，必须在这 n 次试验的原始记录

$$AA\overline{A}A\cdots\overline{A}A\overline{A}$$

中，有 k 个 A，$n-k$ 个 $\overline{A}$，每个 A 有概率 p 而每个 $\overline{A}$ 有概率 $1-p$. 又这 n 次试验独立，所以每个这样的原始记录序列出现的概率是 $p^k(1-p)^{n-k}$. 这样的序列共有 $\binom{n}{k}$ 个. 所以 X 恰好为 k 的概率

$$p_k=\binom{n}{k}p^k(1-p)^{n-k}$$

如果随机变量 X 的概率分布具有以上形式，则称 X 服从**具有参数** (n,p) **的二项分布**. 记为 $X\sim B(n,p)$.

现在，我们可以很容易的算出我们要求的三个概率值分别为 $\frac{5}{16}$($n=5$,

$k=2,p=0.5)$，$\frac{21\ 875}{839\ 808}\left(n=9,k=4,p=\frac{1}{6}\right)$ 和 $\frac{972}{3\ 125}(n=6,k=2,$ $p=0.4)$.

练习 4

计算机模拟本节开始提出的三个问题. 其中抛硬币可以用0－1随机数来模拟；掷骰子可以用 1 到 6 的随机整数来模拟；模球可以用 1 到 5 的随机整数来模拟，以 1,2 为红球，3,4,5 为白球. 抛硬币 5 次作为一次实验，共作 n 次这样的实验，(比如取 $n=1\ 000$)，看 5 个0－1随机数中恰好出现两次 1 的频率是多少，是否与理论值接近？其余两个问题也作类似模拟.

我们现在利用二项分布来导出另一重要的分布：**Poisson 分布**.

考虑在一定时间内某交通路口所发生的事故个数 X，求 X 恰好为 k 的概率. 为方便计，设所观察的这段时间为 $[0,1)$. 取一个很大的自然数 n，把时间段 $[0,1)$ 分为等长的 n 段：

$$I_1=\left[0,\frac{1}{n}\right),I_2=\left[\frac{1}{n},\frac{2}{n}\right),\cdots,I_j=\left[\frac{j-1}{n},\frac{j}{n}\right),\cdots,I_n=\left[\frac{n-1}{n},1\right)$$

现作如下的假定：

1° 在每段 I_j 内，恰发生一次事故的概率，近似地于这段时间的长度成正比，即可取为 $\frac{\lambda}{n}$. 又假定在 n 很大因而 $\frac{1}{n}$ 很小时，在 I_j 这么短暂的时间内，要发生两次或两次以上事故是不可能的. 因此，在 I_j 时段内不发生事故的概率为 $1-\frac{1}{n}$.

2° $I_1,\cdots,I_n$ 各段内是否发生事故是相互独立的.

此时，在 $[0,1)$ 时段内发生的事故数 X 就等于在 n 个时段 $I_1,\cdots,I_n$ 内有事故的时段数，按 1°, 2° 的假设，X 应服从二项分布 $B\left(n,\frac{\lambda}{n}\right)$. 于是，

$$P(X=k)=\binom{n}{k}\left(\frac{\lambda}{n}\right)^k\left(1-\frac{\lambda}{n}\right)^{n-k}$$

严格讲，上式只是一个近似式，因为在假设 1° 中，在每个时间段内发生一次事故的概率只是近似等于 $\frac{\lambda}{n}$，当 $n\to\infty$ 时，就得到确切的答案. 因为当 $n\to\infty$ 时有

$$\binom{n}{k}n^{-k}\to\frac{1}{i!},\qquad\left(1-\frac{\lambda}{n}\right)^n\to\mathrm{e}^{-\lambda}$$

所以

$$P(X=k)=\frac{\lambda^k}{k!}e^{-\lambda}$$

如果一个随机变量 X 的概率分布具有上式的形式，我们就称 X 服从**具有参数 λ 的 Poisson 分布**，记为 $X \sim P(\lambda)$.

练习 5

(1)利用本节的知识来解释本实验练习 3(4)的结果.

(2)从本节的推导我们知道，如果随机变量 X 服从二项分布 $B(n,p)$，当 n 很大而 p 很小时，可以近似的认为 X 服从Poisson分布 $P(np)$. 如果 p 不随着 n 变化时，利用计算机模拟来说明

i)对很大的 n，是否仍可以近似的认为 X 服从 Poisson 分布 $P(np)$.

ii) 当 n 很大时，对 $k=20,30,40,50$，分别近似做出 $kP\left(np+\frac{i-1}{k}\sqrt{n} \leqslant X < np+\frac{i}{k}\sqrt{n}\right)$（其中 i 取一切可能的值）的图形（以 $\frac{i}{k}$ 做为横坐标），从这四张图中你能看出些什么？

6.4　正态分布

练习 6

设 T 是在区间 $[0,1]$ 内均匀分布的随机变量. 让 T 连取 n 个值 $t_1,t_2,\cdots,t_n$，则这 n 个值的平均值应当接近于 0.5，它们的和 $S=t_1+t_2+\cdots+t_n$ 应当接近于 $0.5n$. 但真要去做实验，就会发现 S 并不真正等于 $0.5n$，而是有一个偏差 $d=S-0.5n$.

从区间 $[0,1]$ 取出 n 个随机数 $t_1,t_2,\cdots,t_n$，并计算

$$X=\frac{t_1+t_2+\cdots+t_n-0.5n}{\sqrt{n}}$$

这称为一次实验，得到的 X 称为这次实验的结果. 我们来看随机变量 X 的分布. 做 N 次实验，将所得的结果称为 $X_1,X_2,\cdots,X_N$. 可以预计到 X_i 在 0 附近摆动，离 0 越近的 X_i 越多，离 0 越远的 X_i 越少. 取一个短的区间长 d. 对每个正实数 x，计算出落在区间 $\left[x-\frac{d}{2},x+\frac{d}{2}\right)$ 内的 X_i 的个数 N_x，以 $\frac{N_x}{N_d}$ 作为随机变量 X 在点 x 的概率密度 $f(x)$.（当然应当取 x 离 0 不太远，否则 $f(x)$

可能等于 0.

取 N 和 n 足够大，(比如 $n=100$ 或更大，$N=1\ 000,10\ 000$ 等). 计算出足够多的 $f(x)$，在坐标系中画出所得的数据点 $(x,f(x))$，连成光滑曲线. 观察它好象是什么函数的曲线？令 $\sigma=\left(\sum\limits_{i=1}^{N}X_i^2\right)/N$. 将函数

$$\varphi(x)=\frac{1}{\sqrt{2\pi}\sigma}\mathrm{e}^{-\frac{x^2}{2\sigma^2}}$$

的图像与 $f(x)$ 的图像画在同一坐标系中，观察两个曲线是否吻合.

增大 n 或者减小 n，看曲线有什么变化？实验结果有什么变化？

如果连续型随机变量 X 的概率密度函数具有形式

$$\varphi(x)=\frac{1}{\sqrt{2\pi}\sigma}\mathrm{e}^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

则 X 的分布称正态分布. 正态分布是非常重要的分布. 其中的常数 μ,σ 分别是随机变量 X 的均值和标准差.

6.5 第一反正弦律

在现实中，我们经常会看到这样一种现象，有人在一段时间里的运气连续的很好，而在另一段时间里又连续的不好. 我们将这样一个问题简化为：某人多次的抛掷一枚硬币，如出现国徽朝上的话就给他加一分(运气好)，反之就减一分(运气不好). 以 a_i 记他第 i 次的得分，S_i 记他前 i 次总得分，那么他运气好，就表示 S_i 一直为正，运气不好表示一直为负(零点按前面是大于零或小于零来定). 如果我们假设此硬币是均匀的话，则可能会认为，S_i 大于零和小于零的次数各半.

然而，这是完全错误的，在 20 000 次抛掷中，甲只领先一次的可能性比他领先10 000次的可能性约大 88 倍. 不管抛掷多少次，领先权改变的次数是 0 的可能性最大. 而后依次是 1，2，3，…，这是由第一反正弦律所导出的一个结论.

第一反正弦律 对于而定的 $\alpha(0<\alpha<1)$，S_i 处于正边的总次数与试验总次数之比 $\frac{k}{n}$ 小于 α 的概率在 $n\to\infty$ 时趋于

$$\pi^{-1}\int_0^{\alpha}\frac{\mathrm{d}x}{[x(1-x)]^{\frac{1}{2}}}=2\pi^{-1}\arcsin\alpha^{\frac{1}{2}}.$$

这个逼近在 $n=20$ 时就很好了.

练习 7

利用计算机模拟第一反正弦律所描述的现象,验证其结论.

实验七 几何变换

如果有一种法则 φ，将平面 Π 上每个点 A 对应于唯一的一个点 $\varphi(A)$，则 φ 称为平面上的变换. $\varphi(A)$ 称为 A 的象. 平面上的图形由点组成，因面平面上每个变换 φ 将每个图形 C 变到某个图形 $\varphi(C)$. $\varphi(C)$ 称为 C 的象.

在平面上建立了直角坐标系之后，每个点 P 由它的坐标 (x,y) 来代表，其中 x,y 是一对实数. 平面上的变换 φ 将每个点 $P(x,y)$ 变到点 $\varphi(P)(x',y')$，$\varphi(P)$ 的横坐标和纵坐标都是 x,y 的函数：

$$x' = f_1(x,y), \quad y' = f_2(x,y)$$

只要给定了两个函数 f_1 和 f_2，就决定了一个几何变换 φ，它将坐标为 (x,y) 的点变到坐标为 $(f_1(x,y),f_2(x,y))$ 的点.

设平面曲线 C 的参数方程为

$$x = x(t), \quad y = y(t), \qquad t \in T$$

其中 T 是函数 $x(t),y(t)$ 的定义域. 则曲线 C 在变换 φ 下的象 $\varphi(C)$ 的参数方程为

$$x = f_1(x(t),y(t)), \quad y = f_2(x(t),y(t)), \qquad t \in T$$

德国数学家克莱茵(F. Klein)1872 年在德国爱尔兰根大学的一次学术报告中提出，几何学的任务就是研究在一定的几何变换(这些变换组成一个群)下图形的不变性质. 比如，欧几里德平面几何学里的变换是平面图形的平移、转动、轴对称，在这些变换下，长度、角度保持不变. 克莱茵所阐述的这一著名观点被称为爱尔兰根纲领.

本实验的主要内容是观察在几种重要的几何变换下图形的变化情况，以及在这些变换下图形的哪些性质保持不变.

7.1 线性变换与仿射变换

设变换 φ 将平面上所有的点绕原点旋转 α 角(以逆时针方向为正). 可以这样来求出每个点 $P(x,y)$ 与它的象 $P'(x',y')$ 的坐标之间的函数关系：以原点为极点、x 轴正方向为极轴建立极坐标系. 设 P 点的极坐标是 (r,θ) ，则 P' 点的极坐标为 $(r,\theta + \alpha)$ ，

$$x = r\cos\theta$$
$$y = r\sin\theta$$
$$x' = r\cos(\theta+\alpha) = r\cos\theta\cos\alpha - r\sin\theta\sin\alpha = x\cos\alpha - y\sin\alpha$$
$$y' = r\sin(\theta+\alpha) = r\cos\theta\sin\alpha + r\sin\theta\cos\alpha = x\sin\alpha + y\cos\alpha$$

注意 $\cos\alpha,\sin\alpha$ 都是与 x,y 无关的常数，

$$x' = x\cos\alpha - y\sin\alpha \quad 与 \quad y' = x\sin\alpha + y\cos\alpha$$

都是 x,y 的一次齐次函数，即线性函数.

一般的，如果变换 $\varphi:(x,y)\mapsto(x',y')$ 由函数关系

$$x' = a_1x + b_1y$$
$$y' = a_2x + b_2y$$

决定，其中 a_1,b_1,a_2,b_2 是与 x,y 无关的常数，则称 φ 为线性变换，它也可写成矩阵形式

$$\begin{pmatrix}x'\\y'\end{pmatrix} = \begin{pmatrix}a_1 & b_1\\a_2 & b_2\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}$$

由矩阵

$$A = \begin{pmatrix}a_1 & b_1\\a_2 & b_2\end{pmatrix}$$

决定.

进一步，如果在变换 φ 下每个点 (x,y) 的象 (x',y') 的坐标 x',y' 都是 x，y 的一次函数：

$$x' = a_1x + b_1y + c_1$$
$$y' = a_2x + b_2y + c_2$$

则称 φ 是仿射变换. 仿射变换可以看成是先作一个线性变换、再作一次平移 $(x,y)\mapsto(x+c_1,y+c_2)$ 得到的变换.

练习 1

(1)任选一个角度 α，决定一个线性变换 $\varphi:(x,y)\mapsto(x',y')$ 使

$$x' = x\cos\alpha - y\sin\alpha$$
$$y' = x\sin\alpha + y\cos\alpha$$

画一个由平面直线段或曲线段组成的图形 C，再画出它在上述变换下的象 $\varphi(C)$. 观察 $\varphi(C)$ 是否是由 C 经过旋转得到的图形.

想一想，如果变换 φ 是关于某条直线 $ax+by+c=0$ 的轴对称，则任一点 (x,y) 的象的坐标 (x',y') 与 x,y 有怎样的函数关系？通过画图和观察验证你的结论是否正确.

(2)自己选取变换矩阵 $\boldsymbol{A}=\begin{pmatrix}a_1 & b_1\\ a_2 & b_2\end{pmatrix}$，决定线性变换 φ.

画一条曲线或由曲线段(或直线段)组成的图形 C. 再画出这个图形经过变换 φ 作用之后的图形. 比较变换前后图形的形状和位置，看发生了哪些变化，哪些性质保持不变？为了便于观察图形的变化情况，可画若干条平行于坐标轴的直线 $x=k, y=k$ (其中 $k=0,\pm 1,\pm 2,\cdots$)，组成正方形网格，作为图形 C 的背景. 再画出这些正方形网格以及图形 C 经过变换 φ 之后的象. 观察正方形网格的位置和形状的变化情况，有助于理解和想象图形 C 的变化情况.

特别注意观察以下现象：

直线是否仍变成直线？平行直线是否仍变成平行直线？平行四边形是否仍变成平行四边形？

两点之间的距离经过变换是否保持不变?如果变化了，是否所有的距离按同一比例放大或缩小？或者，同一方向上的距离按同一比例放大成缩小？

角度大小是否保持不变？矩形是否仍变成矩形？

圆变成什么图形？是否仍变成圆？

(注：以上问题都可通过理论推导得出结论. 本实验课的要求是通过对图形的观察来得出结论成验证这些结论.)

一般要求变换矩阵满足条件 $a_1b_2-a_2b_1\neq 0$？如果选取矩阵使 $a_1b_2-a_2b_1=0$，看看发生什么观象？

7.2 线性变换的特征向量

选取一个矩阵 $\boldsymbol{A}=\begin{pmatrix}a_1 & b_1\\ a_2 & b_2\end{pmatrix}$，决定线性变换

$$\varphi:\begin{pmatrix}x\\ y\end{pmatrix}\mapsto\begin{pmatrix}x'\\ y'\end{pmatrix}=\begin{pmatrix}a_1 & b_1\\ a_2 & b_2\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix}$$

平面上每个点 $P(x,y)$ 被变到 $P'(x',y')$，向量 $\overrightarrow{OP}$ 也就被变到 $\overrightarrow{OP'}$. 试通过画图来研究向量的方向的变化情况. 看哪些向量的方向经过变换之后向逆时针方向偏移，哪些向顺时针方向偏移，哪些向量的方向保持不变？

先对矩阵

$$\boldsymbol{A}=\begin{pmatrix}1+\varepsilon_1 & \varepsilon_2\\ \varepsilon_3 & 1+\varepsilon_4\end{pmatrix}$$

($\varepsilon_1,\varepsilon_2,\varepsilon_3,\varepsilon_4$ 是小的正实数)作下面的实验. 再换成别的矩阵作同样的实验.

练习 2

选取自然数 n. 在单位圆周上依次取 n 个点 $P_i\left(\cos\frac{2k\pi}{n},\sin\frac{2k\pi}{n}\right)$（$0\leqslant k\leqslant n-1$）. 设 P_i' 是 P_i 经变换 φ 后的象. 用一种颜色（比如红色）画出从原点 O 到每个 P_i 的线段并适当延长，用另一种颜色（比如绿色或黑色）画出从每个 P_i 到 P_i' 的线段. 观察比较向量 $\overrightarrow{OP_i}$ 与 $\overrightarrow{P_iP_i'}$ 方向的差异. 是否有 $\overrightarrow{OP_i}$ 与 $\overrightarrow{P_iP_i'}$ 方向一致？如果有，有几个？

在线性代数课程中可以知道，如果非零向量 $\boldsymbol{u}=\overrightarrow{OP}$ 在线性变换 φ 的作用下的象 $\overrightarrow{OP'}=\boldsymbol{Au}$ 与 $\boldsymbol{u}$ 的方向平行（方向相同或相反或等于零），即

$$\boldsymbol{Au}=\lambda\boldsymbol{u}$$

对某个数 λ 成立，则 $\boldsymbol{u}$ 称为 φ 的特征向量，λ 称为特征根或特征值. 平面线性变换 φ 的特征值 λ 是一元二次方程

$$\begin{vmatrix}\lambda-a_1 & -b_1\\ -a_2 & \lambda-b_2\end{vmatrix}=0$$

的根.

练习 3

按照练习 2 所说的方式在单位圆上取 n 个点 P_i. 或者在正方形 $\{(x,y)\mid |x|\leqslant1,\ |y|\leqslant1\}$ 内随机选取 n 个点 P_i. 将每个点 P_i 经过线性变换 φ 作用后的象 $\varphi(P_i)$ 记为 P_{i1}. P_{i1} 再经过 φ 作用后的象记为 P_{i2}. 一般地，将 P_i 被 φ 连续作用 k 次之后的象记为 P_{ik}. 在坐标平面上画出所有这 n 个点 P_i（$1\leqslant i\leqslant n$），以及它们经过 φ 一次又一次作用之后的象 $P_{i1},P_{i2},\cdots$. 观察所有这些象的变化趋势. 随着作用次数 k 的增大，所有的点 P_{ik} 是否趋于一条直线？如果是，这条直线就是 φ 的一个特征向量方向，它所对应的特征值是所有特征值中最大的.

将矩阵 $\boldsymbol{A}$ 换成 $\boldsymbol{A}^{-1}$ 重做上面的实验，观察和分析所得的结果.

对各种不同的矩阵 $\boldsymbol{A}$ 进行实验.

练习 3 提供了一种计算特征向量的方法，它不仅适用于平面（即二维空间）的变换，而且适用于一般的 n 维空间的变换. 每个 n 阶方阵

$$\boldsymbol{A}=\begin{pmatrix}a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{pmatrix}$$

决定 n 维空间的一个线性变换 φ，将每个列向量

$$X=\begin{pmatrix}x_1\\x_2\\\vdots\\x_n\end{pmatrix}\quad \text{映到}\quad Y=\begin{pmatrix}y_1\\y_2\\\vdots\\y_n\end{pmatrix}=AX$$

为了书写方便，我们以后将列向量 X 写成 $(x_1,\cdots,x_n)^{\mathrm{T}}$ 的形式，它由行向量 $(x_1,\cdots,x_n)$ 转置而成，$x_1,\cdots,x_n$ 依次是各列的元素.

从某个给定的非零列向量 $X_0=(x_{01},x_{02},\cdots,x_{0n})^{\mathrm{T}}$ 出发，用 φ 不断作用，依次得到

$$X_1=AX_0,\quad X_2=AX_1,\quad X_3=AX_2,\quad \cdots,\quad X_k=AX_{k-1},\quad \cdots$$

我们无法像 $n=2$ 的情况那样画出平面图形来观察 X_k 是否随着 k 的增大而趋于一个固定的方向，但我们希望作数值计算验证是否 X_k 与 X_{k-1} 趋于成比例. 为此，在一开始先将 $X_0=(x_{01},\cdots,x_{0n})^{\mathrm{T}}$ 除以各分量绝对值之和 $c_0=|x_{01}|+|x_{02}|+\cdots+|x_{0n}|$，化为 $Y_0=c_0^{-1}X=(y_{01},y_{02},\cdots,y_{0n})^{\mathrm{T}}$，其中每个 $y_{0i}=c_0^{-1}x_{0i}$，从而 $|y_{01}|+|y_{02}|+\cdots+|y_{0n}|=1$. 这种将向量 X_0 乘以常数 c_0^{-1} 化为 Y_0 使各分量绝对值之和为 1 的过程叫做**归一化**. 取 $X_1=AY_0=(x_{11},x_{12},\cdots,x_{1n})^{\mathrm{T}}$，再对 X_1 作归一化得 $Y_1=c_1^{-1}X_1$（其中 $c_1=|x_{11}|+|x_{12}|+\cdots+|x_{1n}|$）. 照此下去依次得 $Y_0,Y_1,Y_2,\cdots,Y_{k-1},Y_k,\cdots$，其中 $Y_k=c_k^{-1}AY_{k-1}$，c_k 等于 AY_{k-1} 各分量绝对值之和. 如果随着 k 的增大，$Y_{k-1}-Y_k$ 的所有分量都趋于 0，即 Y_k 趋于一个固定的向量 Y，则 $AY=cY$ 对某个常数 c 成立，Y 是特征向量，c 是 A 的绝对值最大的特征值. 这个求特征向量的方法叫做**幂方法**.

练习 4

选定一个矩阵 A，使它的所有的矩阵元素 $a_{ij}\geqslant 0$. 取初始向量 $Y_0=(1/n,\cdots,1/n)^{\mathrm{T}}$ 的所有分量相等，都为 $1/n$. 按上述幂方法进行实验，看 $Y_{k-1}-Y_k$ 是否随着 k 的无限增大而趋于零向量.

7.3 射影变换

练习 5

定义平面上的变换 $\varphi:(x,y)\mapsto(x',y')$ 使 x',y' 之间的函数关系为

$$x'=\frac{x}{x-1},\quad y'=\frac{y}{x-1}$$

画一条曲线或由曲线段(或直线段)组成的图形 C. 再画出这个图形经过变换 φ 作用之后的图形. 比较变换前后图形的形状和位置,看发生了哪些变化,哪些性质保持不变.

特别注意观察:

直线是否仍变成直线? 平行直线是否仍变成平行直线? 同一方向上的距离是否按同一比例放大或缩小?

直线 $x=1$ 上的点没有象. 当点 P 从直线 $x=1$ 之外接近这条直线时,它的象 $\varphi(P)$ 的变化怎样?

在直线 $x=1$ 上任取一点 P,过 P 作若干条不同方向的直线,观察这些直线在变换 φ 下的象,它们是否仍是相交直线? 如果是,相交于何处?

以原点为圆心作若干个圆,它们的半径有的小于 1,有一个等于 1,有的大于 1. 观察这些同心圆被变成什么图形.

练习 5 中的变换不是仿射变换,而是射影变换的一个例子. 一般的,设实系数矩阵

$$\begin{pmatrix}a_1 & b_1 & c_1\\ a_2 & b_2 & c_2\\ a_3 & b_3 & c_3\end{pmatrix}$$

的行列式不等于 0. 定义平面上的将坐标为 (x,y) 的点 P 变到坐标为 (x',y') 的点 P' 的变换 φ,使 x',y' 与 x,y 之间有函数关系

$$x' = \frac{a_1x + b_1y + c_1}{a_3x + b_3y + c_3}$$

$$y' = \frac{a_2x + b_2y + c_2}{a_3x + b_3y + c_3}$$

则 φ 称为射影变换. 如果 $a_3 = b_3 = 0$, $c_3 \neq 0$,则射影变换是一个仿射变换. 可见,仿射变换是射影变换的特例.

想象平面图形 C 是画在一块玻璃板上,用一个点光源将玻璃板上的图形 C 投影到另一个平面上得到图形 C' . 在玻璃板和被投影的平面上分别建立坐标系,则从 C 到 C' 的变换就是射影变换. 一个接一个施行若干个这种投影,总的效果仍然是射影变换.

另外选择一个射影变换,重作练习 5.

7.4 非欧几何

欧几里得几何的一个重要的公理是平行公理:在平面上,过任一条直线 l 外一点 P 能且只能作一条直线与 l 不相交.

将这一公理加以改变,即引入这样的公理:在平面上,过任一条直线 l 外一点 P 至少可以作两条不同的直线与 l 不相交. 这样就可以得到一种非欧几里得几何学,称为罗巴切夫斯基几何,或简称罗氏几何.

罗氏几何看起来与我们的常识相冲突. 但也可以利用欧氏几何建立一些模型帮助我们理解它,知道它与欧氏几何同样是合理的. 利用射影变换可以建立罗氏几何的一种模型如下,称为克莱茵模型.

在欧氏平面上建立直角坐标系. 以原点为圆心、1 为半径画一个圆 G . 圆 G 内部的点的集合作为罗氏几何的整个平面. 直线在圆 G 内部的部分作为罗氏几何的直线. 圆 G 内部的图形 C 称为罗氏几何的图形.

如果一个射影变换 φ 将圆 G 变到自己,则称 φ 是罗氏几何变换. 任一罗氏几何图形(即圆 G 内部的图形) C 在任一罗氏几何变换下的象 $\varphi(C)$ 被认为与 C 全等. 注意,罗氏变换一般并不保持按欧氏几何定义的长度和角度. 但是,我们可以在罗氏几何中重新定义长度和角度,使任一条线段与它在任一罗氏变换下的象的长度相等,任一个角与它在任一罗氏变换下的象角度相等.

欧氏平面上哪些射影变换 φ 可以将圆 G 变到自己?显然,圆 G 绕圆心的旋转以及关于直径的对称都符合这个要求,因而都是罗氏变换的例子,但它们也都是欧氏变换,还不足以体现罗氏几何的特点. 特别是,它们不能将一把"尺子"移到任何位置去量长度. 下面的练习给出了一类非欧氏的罗氏变换.

练习 6

任意取定一个实数 α. 定义欧氏平面上的射影变换 $\varphi:(x,y)\mapsto(x',y')$ 使

$$x'=\frac{x\operatorname{ch}\alpha+\operatorname{sh}\alpha}{x\operatorname{sh}\alpha+\operatorname{ch}\alpha},\quad y'=\frac{y}{x\operatorname{sh}\alpha+\operatorname{ch}\alpha}$$

其中 $\operatorname{ch}\alpha=(e^{\alpha}+e^{-\alpha})/2$ 与 $\operatorname{sh}\alpha=(e^{\alpha}-e^{-\alpha})/2$ 分别是双曲正弦与双曲余弦. 这样的变换 φ 称为**双曲旋转**.

(1)画图验证(也用计算验证):双曲旋转 φ 将圆 $G:x^2+y^2=1$ 变到自身. 因而是罗氏变换.

(2)制作罗氏刻度尺:

设 $X_0(-1,0)$, $X(1,0)$ 分别是 x 轴与圆周 G 的两个交点. 观察(或计算验证)欧氏线段(即罗氏直线) X_0X 被变换 φ 变到自身.

在原点 O 用一条短的竖线标出一个刻度. 画出 O 在 φ 的依次重复作用下的象 $M_1=\varphi(O)$, $M_2=\varphi(M_1)$, $M_3=\varphi(M_2)$, $\cdots$, $M_{k+1}=\varphi(M_k)$, $\cdots$,在所有这些点都用短竖线标出刻度. 所有这些刻度使 OX 组成了一把刻度尺. OM_1 是长度单位,它可以被罗氏变换 φ 依次重复作用变到相邻两刻度之间的线段 OM_1, M_1M_2, M_2M_3, $\cdots$, M_kM_{k+1}, $\cdots$. 因此,所有相邻两刻度之间的罗氏长度相等,(尽管它们的欧氏长度越来越短). 当 φ 作用的次数 k 无限增大时,作用的象 M_k 沿 OX 方向无限接近 X ,但是永远不能到这 X . 这就是说, OX 的长度等于无限多个单位长度 OM_1, M_1M_2, $\cdots$, M_kM_{k+1}, $\cdots$ 之和,趋于无穷大. 也就是说, O 到 X 的距离是无穷远. 将平面绕原点 O 旋转任意角度,也是罗氏变换,保持罗氏距离不变,而上面制作出的刻度尺 OX 可以旋转成 O 朝任意方向到圆周 G 上的一把刻度尺,圆周上 G 所有的点到 O 的距离都是无穷大, G 是无穷远点的集合. 特别, OX 可旋转成与自己的相反方向的刻度尺 OX_0 ,它与 OX 方向的刻度尺共同组成 X_0X 上的刻度尺,刻度由 O 向左右两个方向无限延伸. 这说明 X_0X 确实是两端无限延长的罗氏直线.

将 x 轴方向上的刻度尺旋转为 y 轴方向上的刻度尺,再作出 y 轴方向上的这个刻度尺在上述由 α 决定的双曲旋转 φ 下的象. 改变 α 的值,可以将刻度尺移动到离原点的距离为任意值 $d<1$ 的地方,然后又可以旋转到任意方向. 这样就可以将刻度尺摆到任意地方去度量长度.

罗氏平行公理: 设 l 是罗氏平面内任意一条直线, A 是直线外一点. 从欧氏几何看来, l 是圆 G 上两点 P,Q 之间的线段. 延长直线段 PA 与 G 交于 R ,延长直线段 QA 与 G 交于 S . 则过 A 的两条不同的直线 PR, QS 都与 PQ 不相交(因为它们与 PQ 在欧氏几何下的交点在圆上而不在圆内). 而且,凡是过

A 点且穿过 $\angle PAS$ 内部与 $\angle QAR$ 内部的直线与 PQ 也都不相交.

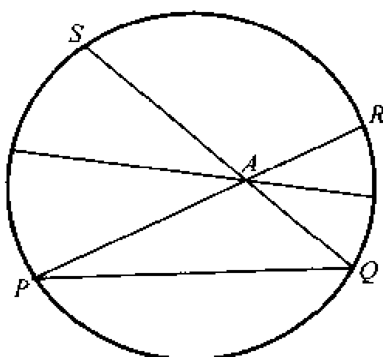


图 7－1

(3)制作罗氏量角器：

任意取定自然数 n，设 x 轴的正方向与 G 的交点为 X. 从 X 开始在圆周 G 上作 n 个点 $X, X_1, X_2, \cdots, X_{n-1}$ 将圆周 n 等分. 则 O 到这些分点所作的 n 条连线将以 O 为顶点的周角分成 n 个在欧氏几何中相等的角，这就做成了一个以 O 为中心的欧氏量角器. 将平面绕 O 作逆时针方向 $2\pi/n$ 的旋转也是罗氏变换，它将上述的 n 个角中的每一个交到下一个. 这说明以 O 为中心的这个欧氏量角器同时也是罗氏量角器.

用本练习开始所定义的双曲旋转将 O 点移动到 M 点，同时也将以 O 为中心的上述量角器变成以 M 为中心的罗氏量角器. 观察看它是否仍是欧氏量角器.

过一点 A 作已知直线 l 的垂线：用罗氏变换 φ_1 将 A 移到原点 O，直线 l 移到 l'. 过 O 作 l' 的欧氏垂线 m'，m' 同时也是 l' 的罗氏垂线. 再用 φ_1 的逆变换将 O, l' 分别变回 A, l，则 m' 变到所需的垂线 m. 试自己完成这个作图，观察 m, l 在欧氏几何下是否一定垂直?

自己研究罗氏几何的其它问题.

7.5 证明代数基本定理

平面上以 (x,y) 为坐标的点可以用一个复数 $x+y\mathrm{i}$ 来表示. 这样，平面就称为复平面，每一个变换 $\varphi:(x,y)\mapsto(x',y')$ 可以看作由一个复变量函数

$$z'=f(z)$$

决定，其中 $z=x+y\mathrm{i}, z'=x'+y'\mathrm{i}$，而 x,y,x',y' 都是实数. 选择不同的复变量函数 f，就可以决定不同的变换. 在叙述的时候，不妨将每个复数 z 与它

所代表的点等同，将复函数 f 与它所导致的变换 $z \mapsto f(z)$ 等同.

练习 7

(1)选定四个复常数 a,b,c,d 满足条件 $ad-bc\neq 0$，决定一个复变换 $f(z)=\dfrac{az+b}{cz+d}$. 画出并观察各种图形在 f 作用下的变化情况. 特别注意直线与圆被 f 变成什么图形，并对你观察到的现象作一理论证明.

(2)取复函数 $f(z)=z^2$. 画出并观察 x 方向和 y 方向的等距平行线组成的网格变成什么图形.

(3)设图形

$$C=\{x \mid x\in\mathbf{R}, |x|\geqslant 1\}\cup\{\cos\theta+\mathrm{i}\sin\theta \mid 0\leqslant\theta\leqslant\pi\}$$

由以原点为圆心的单位圆的上半圆周以及实轴(即 x 轴)在这个单位圆外的部分连接而成. 分别作出它在变换

$$f_1(z)=\frac{z+1}{z-1} \quad 和 \quad f_2(z)=\left(\frac{z+1}{z-1}\right)^2$$

作用下的象. 设法辨认出 C 的三个组成部分

$$C_1=\{x \mid x\in\mathbf{R}, x\leqslant -1\}$$
$$C_2=\{\cos\theta+\mathrm{i}\sin\theta \mid 0\leqslant\theta\leqslant\pi\}$$
$$C_3=\{x \mid x\in\mathbf{R}, x\geqslant 1\}$$

分别变成了哪一部分图形.

利用复变换可以给出代数基本定理的一个证明.

古典的代数的基本任务是解方程. 因此，关于方程是否有根的如下定理就被称为**代数基本定理**：

代数基本定理 复系数一元 n 次方程

$$a_0x^n+a_1x^{n-1}+\cdots+a_{n-1}x+a_n=0$$

至少有一个复数根，(其中 $a_0,a_1,\cdots,a_n$ 是已知的复数，且 $a_0\neq 0$).

这个定理是高斯(C. F. Gauss)发现并证明的.

练习 8

任选一个复系数多项式 $f(z)=a_0z^n+a_1z^{n-1}+\cdots+a_{n-1}z+a_n$，其中 a_0，a_1，…，a_n 是复常数且 $a_0\neq 0$，且不妨设 $a_0=1$. 通过作图来观察为什么方程 $f(z)=0$ 一定有一个根.

取定一个非负实数 r 作为 z 的模. 当复数 z 的幅角 θ 从 0 连续变化到 2π

时，$z=r(\cos\theta+\mathrm{i}\sin\theta)$ $(0\leqslant\theta<2\pi)$ 所对应的点在复平面上画出一个半径为 r 的圆 C_r，而 $f(z)$ $(z=r(\cos\theta+\mathrm{i}\sin\theta),0\leqslant\theta<2\pi)$ 所对应的点在复平面上画出一条封闭曲线 $f(C_r)$. 在特殊情形 $r=0$ 下，C_0 就是原点，$f(C_0)$ 是点 a_n.

注意 $f(z)=x+y\mathrm{i}$ 的实部 x 与虚部系数 y 都是实的自变量 θ 的一元函数. 如果所用的计算机软件不能由复函数表达式 $f(z)$ 直接画出曲线 $f(C_r)$，可以写出 x,y 与 θ 的函数关系表达式，作为曲线 $f(C_r)$ 的参数方程，再由这个参数方程画出图来. 例如，当所有的系数 a_i 都是实数时，

$$x=a_0r^n\cos n\theta+a_1r^{n-1}\cos(n-1)\theta+\cdots+a_{n-1}r\cos\theta+a_n$$

$$y=a_0r^n\sin n\theta+a_1r^{n-1}\sin(n-1)\theta+\cdots+a_{n-1}r\sin\theta$$

对一般的情况 $a_i=a_{i1}+a_{i2}\mathrm{i}$ $(a_{i1},a_{i2}\in\mathbf{R},0\leqslant i\leqslant n)$，读者容易自己写出 x,y 的表达式.

我们的目的是要证明：必然存在一个正实数 r，使得以原点为圆心、r 为半径的圆 C_r 的象 $f(C_r)$ 经过原点 O(即复数 0 所对应的点).

(1)取一个 r，画出曲线 $f(C_r)$，同时也画出圆 C_r 在变换 $f_1(z)=z^n$ 下的象 $f_1(C_r)$.

看一看，想一想，$f_1(C_r)$ 是什么图形？它是否将原点 O 包含在内部？$f(C_r)$ 是否将原点 O 包含在内部？

取很大的 r，看 $f_1(C_r)$ 与 $f(C_r)$ 的图象是否相当接近？$f(C_r)$ 是否将原点包含在内？试对你所观察到的现象作一理论解释.

可以复现，无论对于什么多项式 f，只要 r 取足够大的值 R，就可以使 $f(C_R)$ 与 $f_1(C_R)$ 的图象相当接近，使它们都将原点包含在内.

(2)将 r 从 R 值逐渐缩小，观察 $f(C_r)$ 的变化趋势.

由于 $f(C_R)$ 将原点 O 包含在内，而 $r=0$ 所对应的图象 $f(C_0)$ 是一个点 a_n. 当 r 从 R 连续减小到 0 时，$f(C_R)$ 连续变化，最后收缩为一个点，在收缩过程中必然要扫过原点 O. 也就是说，在 R 与 0 之间必然有某一个 $r=r_0$ 值使 $f(C_{r_0})$ 经过原点，C_{r_0} 上有一点 $z_0=r_0(\cos\theta_0+\mathrm{i}\sin\theta_0)$ 的象 $f(z_0)=0$. 这个事实对于所有的多项式都成立，这就说明了代数基本定理的正确性.

当然，作图只能提供一个直观的思路，还需要将这个思路用严格的数学语言叙述出来，才能算是对代数基本定理的一个真正的证明.

(3)搜索复根：

实际作图时，如果能用动画的形式让 $f(C_R)$ 连续变化到 $f(C_0)$，可以直接观察到 $f(C)$ 扫过原点的现象.

否则，只能让 r 取若干离散的值 $r_1,r_2,\cdots$ 从 R 减少到 0，观察曲线 $f(C)$

越过原点的现象. 如果封闭曲线 $f(C_{r_k})$ 的所有的圈都将原点包含在内，而 $f(C_{r_{k+1}})$ 已经有某一圈越过了原点，则 $f(C_r)$ 在 r_k 与 r_{k+1} 之间某个值 r 时刚好扫过原点. 此时取 r_k 与 r_{k-1} 的平均值 $r_{k+1/2} = (r_k + r_{k+1})/2$ ，看 $f(C_{r_{k+1/2}})$ 是否已越过原点. 如果没有越过，则 $f(C_r)$ 在某个 $r_{k+1/2} < r < r_{k+1}$ 处扫过原点，取 $r_{k+1/2}$ 与 r_{k+1} 的平均值再试. 如果 $f(C_{r_{k+1/2}})$ 已越过原点，则取 r_k 与 $r_{k+1/2}$ 的平均值再试. 重复此过程，可求得具有足够精确度的 r_0 ，使 $f(C_{r_0})$ 近似地经过原点. 这就求得了一个复根 z_0 的模 r_0 .

下面还需要求出幅角 θ_0 使 $z_0 = r_0(\cos\theta_0 + \mathrm{i}\sin\theta_0)$ 满足条件 $f(z_0) = 0$. 设 $f(r_0(\cos\theta + \mathrm{i}\sin\theta)) = x + y\mathrm{i}$ ，其中 $x, y \in \mathbf{R}$ 都是 θ 的函数. 在同一坐标系中画出

$$x = x(\theta),\quad y = y(\theta),\qquad 0 \leqslant \theta \leqslant 2\pi$$

的图象，通过观察可找到这两条曲线与 θ 轴的公共交点，取交点的横坐标 $\theta = \theta_0$. 则 $z_0 = r_0(\cos\theta_0 + \mathrm{i}\sin\theta_0)$ 是所求的一个根.

实验八　天体运动

开普勒根据他同时代的天文学家第谷·布拉赫对行星运动长期大量观测的数据，以及他自己对行星运动的观测和巨大的复杂计算，得出了行星运动的三大定律：

第一定律：行星运动的轨道是椭圆，太阳位于这个椭圆的一个焦点.

第二定律：从太阳到行星的所引的矢径在相同时间内扫过相同的面积.

第三定律：行星绕太阳的周期的平方与其轨道半长轴的立方成正比.

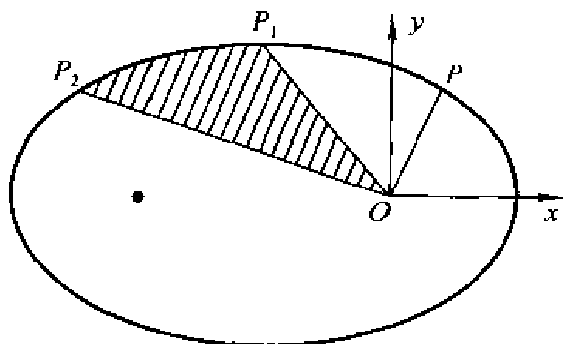


图 8－1

牛顿由开普勒定律导出了万有引力定律. 反过来，由牛顿万有引力定律和牛顿第二定律可以推导出在太阳引力下的天体的运动定律. 开普勒定律和牛顿定律之间的相互推导可以由解微分方程来实现. 本实验则利用数值分析方法来作它们的相互推导.

8.1　行星运动的加速度

我们希望知道行星在任一位置所受力的方向和大小，并研究这个力与行星所在位置的关系. 根据牛顿第二定律，只须根据开普勒定律研究行星运动时的加速度的变化情况，也就知道了行星运动时的受力情况.

实验原理

将行星看作一个质点，以太阳的位置 O 为原点，在行星运行的椭圆平面内建立直角坐标系，写出椭圆方程. 行星在任一时刻 t 的位置 P 用坐标 (x,y)

表示. P 的位置随 t 而变化，它的两个坐标 $x = x(t)$，$y = y(t)$ 都是 t 的函数. 行星的运动可以分解为 x 方向上的分运动与 y 方向上的分运动的合运动. 这两个分运动分别由函数 $x = x(t)$ 和 $y = y(t)$ 来描述.

行星由点 $P_1(x(t_1), y(t_1))$ 运动到点 $P_2(x(t_2), y(t_2))$ 的时间 $\Delta t = t_2 - t_1$ 与矢径 $\overrightarrow{OP_1}$ 到 $\overrightarrow{OP_2}$ 所扫过的椭圆内的部分的面积 $S(P_1, P_2)$ 成正比. 设行星绕椭圆运行一圈的周期为 T，椭圆面积为 S，则 $\Delta t = kS(P_1, P_2)$，其中 $k = T/S$. 因此，P 的位置 (x, y) 随 t 变化的函数关系是已知的，描述两个分运动的函数 $x = x(t)$，$y = y(t)$ 也是已知的. 只要求出这两个分运动在任一时刻 t 的加速度 $\alpha_x(t)$，$\alpha_y(t)$，也就是求出了行星运动的加速度矢量 $\boldsymbol{\alpha}(t)$ 在 x 方向和 y 方向上的分量，$\boldsymbol{\alpha}(t)$ 也就知道了. 研究加速度矢量 $\boldsymbol{\alpha}(t)$ 的方向和大小与行星位置的关系，就可以得出万有引力定律.

怎样求在任一时刻 $t = t_0$ 加速度的两个分量 α_x 和 α_y？以求 α_x 为例：在 t_0 前后很短的一段时间里，可以近似地将 $x = x(t)$ 所描述的运动看作匀加速运动，求出它的加速度作为 $t = t_0$ 时的加速度. 按照高中物理知识，具体求法可为：设 $t_{-1} < t_0 < t_1$，$t_1 - t_{-1}$ 很小，并且已经分别知道 $t = t_{-1}, t_0, t_1$ 时的 x 值 x_{-1}, x_0, x_1. 则从 t_{-1} 到 t_0 这段时间内的平均速度

$$v_{-0.5} = \frac{x_0 - x_{-1}}{t_0 - t_{-1}}$$

它就是在时刻 $t_{-0.5} = (t_{-1} + t_0)/2$ 的瞬时速度. 同样，从 t_0 到 t_1 的平均速度

$$v_{0.5} = \frac{x_1 - x_0}{t_1 - t_0}$$

它就是在时刻 $t_{0.5} = (t_0 + t_1)/2$ 的瞬时速度. 于是加速度

$$\alpha_x(t_0) = \frac{v_{0.5} - v_{-0.5}}{t_{0.5} - t_{-0.5}}$$

同样可求出加速度在 y 方向上的分量 $\alpha_y(t_0)$.

上面所求的加速度分量 $\alpha_x(t_0)$，$\alpha_y(t_0)$ 就是函数 $x(t)$ 和 $y(t)$ 在 $t = t_0$ 对 t 的二阶导数. 上面所说的方法实际上就是：在 $t = t_0$ 的附近的一个很短的区间内，将二阶导数认为是常数，将函数近似地看作不超过二次的多项式函数 $x(t) = k_0 + k_1 t + k_2 t^2$. 已知 $x_i = x(t_i)$（$i = -1, 0, 1$）就可以求出系数 k_0, k_1, k_2，从而求出其二阶导数 $\alpha = 2k_2$ 作为 $t = t_0$ 时二阶导数的值.

实验方法

设行星轨道椭圆半长轴和半短轴分别为 a, b，半焦距为 $c = \sqrt{a^2 - b^2}$，行星运动周期为 T. 设太阳位置为定点 O，行星位置为动点 P.

（1）以太阳位置 O 为原点建立直角坐标系，使椭圆的中心坐标为 $(-c, 0)$. 写出椭圆的参数方程

$$\begin{cases} x = a\cos\varphi - c \\ y = b\sin\varphi \end{cases} \qquad 0 \leqslant \varphi < 2\pi$$

(2)假定行星从 $\varphi = 0$ 的点 $(a-c, 0)$ 开始沿逆时针方向(即 φ 增加的方向)运动.按行星运动的先后顺序在椭圆上依次取 N 个点(比如 $N = 100$) $P_i(x_i, y_i)$ ($i = 0,1,2,\cdots,N-1$),对应的参数 $0 = \varphi_0 < \varphi_1 < \cdots < \varphi_i < \cdots < \varphi_{N-1} = 2\pi$,使所有的 $\varphi_{i+1} - \varphi_i$ 都很小,每两个相邻的点 P_i, P_{i+1} 非常接近.(约定 $\varphi_N = 2\pi$, $P_N = P_0$.)

(3)用三角形 $\triangle P_iOP_{i+1}$ 的面积 ΔS_i 近似地代表矢径 $\overrightarrow{OP_i}$ 到 $\overrightarrow{OP_{i+1}}$ 所扫过的面积.

$$\Delta S_i = \frac{1}{2}\begin{vmatrix} x_i & y_i \\ x_{i+1} & y_{i+1} \end{vmatrix} = \frac{1}{2}(x_iy_{i+1} - x_{i+1}y_i)$$

根据开普勒第二定律,行星从 P_i 到 P_{i+1} 所花的时间 Δt_i 与 S_i 成正比.行星运行一圈的时间等于周期 T,矢径所扫过的面积就是整个椭圆的面积 $S = \pi ab$,取常数 $\mu = T/\pi ab$,则 $\Delta t_i = \mu\Delta S_i$.

(4)将行星从 P_{i-1} 到 P_{i+1} 的运动看作匀加速运动,根据 P_{i-1}, P_i, P_{i+1} 三点的坐标及 $\Delta t_{i-1}, \Delta t_i$ 计算出加速度的 x 分量和 y 分量,作为行星在 P_i 点的加速度分量,从而确定了行星在 P_i 点的加速度矢量.

练习 1

(1)给出椭圆半长轴、半短轴 a, b 的具体值.选定一个固定的常数 k,按照开普勒第三定律 $T^2 = ka^3$ 算出周期 T 的值.按上述实验方法,计算行星在每个位置 P_i 的加速度矢量 $\alpha(P_i)$.

(2)对每一点 P_i,加速度矢量 $\alpha(P_i)$ 的方向与位置矢量 $\overrightarrow{OP_i}$ 的方向有什么关系?

(3)对每一点 P_i,研究两个矢量的大小 $\alpha = \|\alpha(P_i)\|$ 与 $r = \|\overrightarrow{OP_i}\|$ 之间的函数关系 $\alpha = f(r)$.为此,按照你所得到的数据作出函数 $\alpha = f(r)$ 的图像,猜测它是什么函数的图像.再作出 $y = \ln\alpha$ 与 $w = \ln r$ 之间的函数关系 $y = f_1(w)$ 的图像,通过观察图像得出它们之间的函数关系,从而得出 α 与 r 之间的函数关系.你得出了什么结论?计算 αr^2 的值,看它是否随 r 而变动?是否因 a, b 的不同而变动?

讨论和推广

本实验求函数 $x = x(t)$ 在给定的点的二阶导数的方法称为**数值微分法**.一般的说,要求函数 $y = f(x)$ 在给定点 $x = x_0$ 的导数,可以按导数的定义,

取 x 的一个很小的增量 δ，计算差商

$$\frac{f(x_0+\delta)-f(x_0)}{\delta}$$

作为导数的近似值. 这实际上是将函数在区间 $[x_0, x_0+\delta]$ 内近似地当做一次函数来求它的导数. 如果希望更准确一些，取 $x_{-1}<x_0<x_1$ 使 x_1-x_0 和 x_0-x_{-1} 都很小而且比较接近，将函数在区间 $[x_{-1}, x_1]$ 内近似地当做不超过二次的多项式函数 $y=k_2x^2+k_1x+k_0$，由函数在 x_{-1}，x_0，x_1 三点的值求出待定的系数 k_0，k_1，k_2，然后就可求出 x_0 点的一阶导数 $2k_2x_0+k_1$ 和二阶导数 $2k_2$. 如果希望进一步提高精确度，可以在更多的点用更高次的多项式函数去拟合 $y=f(x)$.

8.2 万有引力作用下的天体运动

在练习 1 中，你当了一次牛顿，由开普勒定律得出结论：行星运动时受到的力的方向指向太阳，力的大小与行星到太阳的距离的平方成反比. 这个力就是太阳对行星的万有引力. 现在反过来，已经知道万有引力定律，我们来考察在太阳的引力作用下的天体（包括行星、慧星等）的运动规律.

实验原理和方法

以太阳位置 O 为原点建立平面直角坐标系. 由万有引力定律，天体在任一位置 $P(x,y)$ 的加速度矢量 $\boldsymbol{\alpha}$ 方向指向太阳，其大小 $\alpha=\|\boldsymbol{\alpha}\|$ 与天体到太阳的距离 $r=\|\overrightarrow{OP}\|=\sqrt{x^2+y^2}$ 的平方成反比. 即

$$\boldsymbol{\alpha}=-\frac{k}{r^2}\cdot\frac{\overrightarrow{OP}}{r}$$

其中 k 是与天体位置 P 无关的常数. 由此可以算出 $\boldsymbol{\alpha}$ 的坐标（即它的 x 分量和 y 分量）

$$\alpha_x=-\frac{kx}{r^3},\quad \alpha_y=-\frac{ky}{r^3}$$

任意给定天体的初始位置 $P_0(x_0,y_0)$ 以及初速度矢量 $\boldsymbol{v}_0$ 的坐标 (v_{0x},v_{0y}). 由上面的公式算出天体在这一点的加速度矢量 $\boldsymbol{\alpha}_0$ 的坐标

$$\alpha_{0x}=-kx_0(x_0^2+y_0^2)^{-3/2},\quad \alpha_{0y}=-ky_0(x_0^2+y_0^2)^{-3/2}$$

选定一段很短的时间段 δ. 在这段很短的时间内，可以近似地认为加速度 $\boldsymbol{\alpha}_0$ 不变（实际上是变化很小）. 天体以这个不变的加速度运动，就好比地球上斜抛物体的运动一样. 根据其初位置、初速度、加速度可以算出经过时间 δ 之

后天体的位置 P_1 的坐标 (x_1, y_1) 及速度 $\boldsymbol{v}_1$ 的坐标 (v_{1x}, v_{1y})：

$$\begin{cases} x_1 = x_0 + v_{0x}\delta + \dfrac{1}{2}\alpha_{0x}\delta^2 \\ y_1 = y_0 + v_{0y}\delta + \dfrac{1}{2}\alpha_{0y}\delta^2 \end{cases} \quad \begin{cases} v_{1x} = v_{0x} + \alpha_{0x}\delta \\ v_{1y} = v_{0y} + \alpha_{0y}\delta \end{cases}$$

现在以 P_1 为新的出发点，$\boldsymbol{v}_1$ 为新的初速度.用前面同样的方法又可计算出天体在 P_1 点的加速度矢量，以及经过时间 δ 之后的位置 P_2 和速度矢量 $\boldsymbol{v}_2$ 的坐标.

重复上述过程，就可以得出行星运动的一系列的位置 P_i 和在这些位置的速度矢量 $\boldsymbol{v}_i$.依次将这些位置连成一条折线，就近似地得到了行星运动的轨道曲线.(也可将每两个相邻位置 P_i, P_{i+1} 按"斜抛物体"运动轨道用抛物线连接起来，由一段一段抛物线连成行星运动轨道.)

上述求行星运动轨道的方法显然是一个近似方法.每次的时间段 δ 取得越小，精确度越高，但计算量也越大.因此，要根据需要选取适当的时间段.

练习 2

(1)给定天体的不同的初位置和初速度，按照上述方法，求出行星轨道上一系列点的位置的坐标及在这些位置的速度矢量的坐标.

(2)根据你得到的数据，画出天体运动的轨道图线.改变所取的时间段 δ，看所得到的图线有何变化.改变初速度的方向和大小，观察图线的变化情况.轨道图线是否一定是椭圆？有何规律？

(3)根据你得到的数据，验证开普勒第二定律.

讨论与推广

本练习所做的是：已知天体在任一位置 $P(x, y)$ 的位置矢量 $\boldsymbol{r} = \overrightarrow{OP}$ 与加速度矢量 $\dfrac{\mathrm{d}^2\boldsymbol{r}}{\mathrm{d}t^2}$ 之间满足微分方程

$$\frac{\mathrm{d}^2\boldsymbol{r}}{\mathrm{d}t^2} = -\frac{k}{r^2} \cdot \frac{\boldsymbol{r}}{r}$$

(其中 $r = \|\boldsymbol{r}\| = \sqrt{x^2 + y^2}$)，即

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = -kx(x^2 + y^2)^{-3/2}$$

$$\frac{\mathrm{d}^2y}{\mathrm{d}t^2} = -ky(x^2 + y^2)^{-3/2}$$

并且给定了天体在某一初始时刻 t_0 的位置 $P_0(x(t_0), y(t_0))$ 及速度 $\boldsymbol{v}_0 = (x'(t_0), y'(t_0))$，求天体在任一时刻 t 的位置 $P(x(t), y(t))$.

本练习是用数值方法求这个方程的近似解.其基本思路是:在自变量 t 的一个很短的区间 $[t_0, t_0+\Delta t]$ 内,将二阶导数 $\frac{d^2x}{dt^2}, \frac{d^2y}{dt^2}$ 近似地看作不变,由 x,y 及其一阶导数在 $t=t_0$ 点的值求得它们在 $t=t_0+\Delta t$ 点的值.

这个思路可以用来解别的微分方程及其应用问题.比如,解一般的一阶微分方程

$$\frac{dy}{dx} = f(x,y)$$

可从给定的初始点 (x_0, y_0) 出发,取 x 的一个小的增量 $\delta > 0$,认为导数 $\frac{dy}{dx}$ 的值在 x 的小区间 $[x_0, x_0+\delta]$ 内近似地不变,约等于 $f(x_0, y_0)$,因而在这个区间内

$$y(x) = y_0 + f(x_0, y_0)(x - x_0), \quad \forall\ x_0 \leqslant x \leqslant x_0 + \delta$$

取 $x_1 = x_0 + \delta, y_1 = y(x_1) = y_0 + f(x_0, y_0)\delta$.再以 (x_1, y_1) 为新的初始点,用 $f(x_1, y_1)$ 近似地作为在 x 的区间 $[x_1, x_1+\delta]$ 内导数 $\frac{dy}{dx}$ 的值,算出在这个区间内任一 x 所对应的 y 值

$$y(x) = y_1 + f(x_1, y_1)(x - x_1)$$

重复此过程,可以得到在给定范围内 x 所对应的 y 值.这个过程实际上是将函数 $y=y(x)$ 看作分段的一次函数,用一段段很短的直线段组成的折线来逼近 $y=y(x)$ 的曲线.解微分方程的这个数值方法叫做**欧拉折线法**.

8.3　其他几个物理理象的数学模拟

1.点电荷组的电力线及等势面

练习 3

(1)两个等量异种点电荷分别放在 A,B 两点.试在 A,B 所在的平面内画出它们所产生的电场过一些给定点的电力线和等势线.

(2)假定(1)中的点电荷带电量不同,或者带同种电荷,或者有三个点电荷(带电种类及带电量自己设定),在平面内画出所产生的电场的电力线及等势线.

提示:从某一点 P_0 出发,算出这一点的电力线方向,沿着所算出的方向走一小段到 P_1 点.再重新算出 P_1 的电力线方向,沿这个方向再走一小段.重复此过程,画出电力线.用同样的方法画出等势线.

2.凹面镜的正确形状

按照中学物理课程所说，反射面为球面的凹面镜将平行于主轴的光线反射之后会聚于一点，称为这个凹面镜的**焦点**.下面通过实验来验证这个结论.

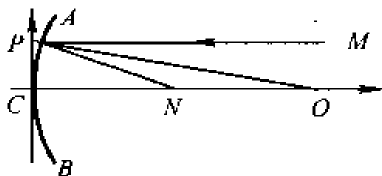


图 8-2

练习 4

(1)设 CO 是凹面镜的主轴，即镜面所在的球面的一条直径，O 是球心，C 是镜面的顶点.过 CO 作轴截面交镜面于一段圆弧 ACB，则镜面可以认为是弧 AC 绕 CO 旋转一周得到.按照凹面镜的要求，弧 AC 所对的圆心角 $\angle AOC$ 应当比较小，比如不超过 $5°$.只须验证这个轴截面上平行于主轴 CO 的光线经过反射后是否会聚于一点.

设 P 是弧 AB 上任一点.过 P 作 $PM \parallel CO$，光线沿 MP 射到镜面上 P 点.作 PN 使 $\angle NPO = \angle OPM$（使 PO 是 $\angle NPM$ 的平分线）.则 PN 是 MP 的反射光线.对弧 AB 上若干个不同的 P 点作反射光线，看它们是否会聚于一点？如果是，会聚于哪一点？

保持顶点 C 和球心 O 不变，将 $\angle AOC$ 增大（比如超过 $45°$ 甚至接近 $90°$），重做上面的实验，看平行于主轴 CO 照射到镜面上的光线是否会聚于一点？

实验表明：当 $\angle AOC$ 很小时，平行光线近似地会聚于 CO 的中点 F.当 $\angle AOC$ 增大时，这一结论的误差越来越大.（实际上，这一结论也很容易由平面几何的理论推导得出.）

既然如此，我们希望知道：当镜面 AOB 成什么形状时，平行于主轴 CO 的光线才真正地会聚于一点？

(2)取一条直线 CF 作为凹面镜的主轴，C 作为镜面的顶点，F 作为焦点.通过作图方法得出过 C 点的镜面的正确形状，使平行于主轴 CF 的光线经镜

面反射后全部会聚于 F 点. 作图步骤如下：

延长 CF 到 O 使 $FO=CF$，即 F 是 CO 的中点. 以 O 为圆心、OC 之长为半径作一小段圆弧 CP，(比如，使 $\angle COP$ 不超过 $5°$). CP 可以近似地作为镜面的一部分，使平行于 CF 且照到这一部分镜面 CP 上的光线会聚于 F. 过 P 作 $PM \parallel CF$. 要使光线 MP 的反射光线为 PF，镜面在 P 点的法线方向 PN 应是 $\angle MPF$ 的角平分线. 过 P 点作 $p \perp PN$，则 p 是镜面在 P 点的切线. 从 P 出发、在 P 上向远离主轴的方向取一条很短的线段(长度由你自己选定) PP_1，近似地作为镜面的一部分. 再用 P_1 点代替 P，用同样的方法确定镜面在 P_1 的切线，在此切线上向远离主轴的方向上截取很短的线段 P_1P_2 作为镜面在 P_1 点的一部分. 重复此过程一直到你愿意停止，作出一条折线，与圆弧 CP 连在一起，近似地作为镜面与轴截面的交线，而它绕主轴旋转所得的曲面作为镜面.

观察你所作出的镜面与轴截面的交线好象是什么曲线？它在靠近顶点 C 的部分近似地是以 O 为圆心的圆弧，而在以后就逐渐远离这个圆. 观察它好象是抛物线. 过你所得的镜面截线的终点作一条以 C 为顶点、以 CF 为对称轴的抛物线，观察它与你所得的镜面截线是否吻合？

试对凸透镜的聚光性能作类似的讨论.

3. 悬链线

将一根绝对柔软、均匀、无伸缩性的细线两端固定，让它自然下垂所成的曲线称为悬链线. 悬链线有什么性质？它的方程是什么？

不妨设悬链线两端的固定点的高度相同，则线的中点 C 为悬低点，称为悬链线的顶点. 以 C 为原点，竖直向上的方向为 y 轴正方向，在曲线所在的平面内建立直角坐标系.

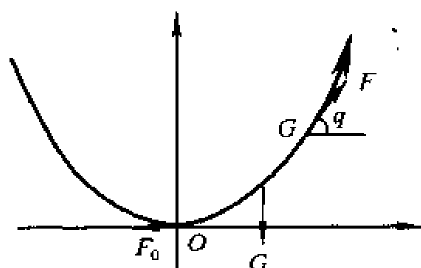


图 8—3

设 P 是曲线上任一点，我们来分析从 C 到 P 这段曲线的受力情况. 由于

处于平衡状态，这段曲线所受合力为 0. 另一方面，这段曲线共受三个力：重力 G，在 C 点所受的水平拉力 F_0，在 P 点所受的倾斜拉力 F. 水平拉力 F_0 与 P 的位置无关，是恒力，设其大小为 f. 倾斜拉力的方向是曲线在 P 点的切线方向，设它与 x 轴正方向的夹角为 q. 重力 G 方向向下，其大小与曲线弧 CP 的长度 s 成正比，可记为 ρs. 如果 P 在第一象限，则 F_0 与 G 的合力 F_1 的坐标为 $(-f, -\rho s)$. $F = -F_1$ 的坐标为 $(f, \rho s)$，$\tan q = \rho s/f = ks$ 与弧长成正比. $\tan q$ 也就是曲线在 P 点的斜率，与弧长 CP 成正比. 曲线在第二象限的部分与它在第一象限的部分关于 y 轴对称.

练习 5

建立直角坐标系. 选定比例系数 a，使悬链线上任一点 P 的斜率 $k_P = as$，其中 s 是由悬链线的顶点到 P 的弧长.

以原点 O 作为悬链线的顶点. 选定一个短的长度 λ. 从 O 向 x 轴正方向上截取长为 λ 的线段 OP_1 作为悬链线的一部分. 悬链线在 P_1 的斜率 k_1 应为 $a \cdot |OP_1| = a\lambda$. 由 P_1 作斜率为 $a\lambda$、长为 λ 的线段 P_1P_2 作为悬链线的下一部分. 再由 P_2 作斜率为 $2a\lambda$、长为 λ 的线段 P_2P_3. 重复这一过程可以作出从 O 出发的一条折线 $OP_1P_2\cdots P_n\cdots$，使组成折线的每条线段 $P_{n-1}P_n$ 的斜率为 $na\lambda$，而长度都是 λ. 这条折线可以认为是悬链线在一象限内的近似形状. 它关于 y 轴的对称图形是悬链线在二象限内的近似形状.

悬链线样子看起来有点象抛物线. 事实上，在顶点附近，曲线的斜率接近于 0，弧长 OP 近似地等于 P 点的横坐标 x. 斜率 $y' = ax$ 正是抛物线 $y = 0.5ax^2$ 的特征性质. 但当 P 离 O 比较远时这一结论就不对了. 在你作的悬链线上取一点 P 离 O 尽可能远. 过 P 作以 O 为顶点、y 轴为对称轴的抛物线，观察它与你所作的悬链线是否比较吻合？选择双曲余弦函数 $y = b\mathrm{ch}(x/b) - b$ 参数 b 使它的图像经过 P 点，观察它的图像与悬链线是否比较吻合？

附录　用微分方程实现开普勒定律和万有引力定律的相互推导

用极坐标 (r, φ) 来表示每一点 P 的位置. 如果以极点 O 为原点、极轴为 x 轴的正方向，则该点的直角坐标为 $(r\cos\varphi, r\sin\varphi)$. 并且可以用复数 $r(\cos\varphi + \mathrm{i}\sin\varphi)$ 来表示点 P 的位置向量 $\boldsymbol{r} = \overrightarrow{OP}$. 记 $\boldsymbol{r}^0 = \cos\varphi + \mathrm{i}\sin\varphi$ 为 $\boldsymbol{r}$ 方向上的单位向量，$\boldsymbol{p}^0 = \mathrm{i}\boldsymbol{r}^0 = -\sin\varphi + \mathrm{i}\cos\varphi$ 为将 $\boldsymbol{r}^0$ 沿逆时针方向旋转直角得到的单位向量. 则 $\boldsymbol{r}^0, \boldsymbol{p}^0$ 构成二维空间的一组基. 特别 $\boldsymbol{r} = r\boldsymbol{r}^0$.

天体的位置向量 $\boldsymbol{r} = r\boldsymbol{r}^0$ 随时间 t 而变化，r,φ 都是 t 的函数. 将 t 的任一函数 x（纯量或向量）对 t 的一阶导数和二阶导数分别用 $\dot{x},\ddot{x}$ 表示. 注意

$$\dot{\boldsymbol{r}}^0 = (-\sin\varphi + \mathrm{i}\cos\varphi)\dot{\varphi} = \mathrm{i}\,\dot{\varphi}\boldsymbol{r}^0 = \dot{\varphi}\boldsymbol{p}^0$$

$$\dot{\boldsymbol{p}}^0 = \mathrm{i}\,\dot{\boldsymbol{r}}^0 = \mathrm{i}^2\dot{\varphi}\boldsymbol{r}^0 = -\dot{\varphi}\boldsymbol{r}^0$$

于是速度向量

$$\boldsymbol{v} = \dot{\boldsymbol{r}} = \dot{r}\boldsymbol{r}^0 + r\dot{\varphi}\boldsymbol{p}^0$$

加速度向量

$$\begin{aligned}\boldsymbol{\alpha} = \ddot{\boldsymbol{r}} &= (\ddot{r}\boldsymbol{r}^0 + \dot{r}\dot{\varphi}\boldsymbol{p}^0) + (\dot{r}\dot{\varphi} + r\ddot{\varphi})\boldsymbol{p}^0 - r\dot{\varphi}^2\boldsymbol{r}^0 \\ &= (\ddot{r} - r\dot{\varphi}^2)\boldsymbol{r}^0 + (2\dot{r}\dot{\varphi} + r\ddot{\varphi})\boldsymbol{p}^0 \\ &= (\ddot{r} - r\dot{\varphi}^2)\boldsymbol{r}^0 + \frac{1}{r}\frac{\mathrm{d}}{\mathrm{d}t}(r^2\dot{\varphi})\boldsymbol{p}^0\end{aligned}$$

注意矢径扫过的面积 S 的微分 $\mathrm{d}S = \frac{1}{2}r^2\mathrm{d}\varphi$. 如果已经知道了开普勒第二定律，即矢径扫过面积的速度 $\frac{\mathrm{d}S}{\mathrm{d}t} = \frac{1}{2}r^2\dot{\varphi}$ 是常量，从而 $h = r^2\dot{\varphi}$ 是常量，$\frac{\mathrm{d}(r^2\dot{\varphi})}{\mathrm{d}t} = 0$，加速度矢量

$$\boldsymbol{\alpha} = (\ddot{r} - r\dot{\varphi}^2)\boldsymbol{r}^0$$

与位置矢量 $\boldsymbol{r}$ 共线. 反过来，如果已经知道 $\boldsymbol{\alpha}$ 与 $\boldsymbol{r}$ 共线（比如牛顿万有引力定律就是这样说的），则 $\frac{\mathrm{d}(r^2\dot{\varphi})}{\mathrm{d}t} = 0$，于是 $h = r^2\dot{\varphi}$ 是常量，开普勒第二定律成立.

以下设 $h = r^2\dot{\varphi}$ 是常量. 我们来计算 α 表达式中的系数 $\ddot{r} - r\dot{\varphi}^2$. 其中 $r\dot{\varphi}^2$ 可写成 $\frac{h^2}{r^3}$. 令 $u = \frac{1}{r}$，则 $r\dot{\varphi}^2 = h^2u^3$. 而

$$\dot{r} = \frac{\mathrm{d}}{\mathrm{d}t}\frac{1}{u} = -\frac{1}{u^2}\frac{\mathrm{d}u}{\mathrm{d}\varphi}\dot{\varphi} = -h\frac{\mathrm{d}u}{\mathrm{d}\varphi}$$

$$\ddot{r} = \frac{\mathrm{d}}{\mathrm{d}t}\left(-h\frac{\mathrm{d}u}{\mathrm{d}\varphi}\right) = -h\frac{\mathrm{d}^2u}{\mathrm{d}\varphi^2}\dot{\varphi} = -h^2u^2\frac{\mathrm{d}^2u}{\mathrm{d}\varphi^2}$$

于是

$$\ddot{r} - r\dot{\varphi}^2 = -h^2u^2\left(\frac{\mathrm{d}^2u}{\mathrm{d}\varphi^2} + u\right)$$

现在设开普勒三定律成立. 行星运动轨道是椭圆，半长轴、半短轴分别是 a,b. 可设极点是其焦点，其方程为

$$r = \frac{p}{1 + e\cos\varphi}$$

其中 $e = \frac{c}{a}$ 是偏心率，$p = \frac{b^2}{a}$. 此时

$$u=\frac{1}{r}=\frac{1+e\cos\varphi}{p}$$

$$-h^2u^2\left(\frac{\mathrm{d}^2u}{\mathrm{d}\varphi^2}+u\right)=-h^2u^2\left(\frac{-e\cos\varphi}{p}+\frac{1+e\cos\varphi}{p}\right)=-\frac{h^2u^2}{p}$$

$$\alpha=-\frac{k}{r^2}\boldsymbol{r}^0$$

其中 $k=\frac{h^2}{p}$ 是与 φ 无关的常数.由于 $\frac{1}{2}h=\frac{1}{2}r^2\dot{\varphi}$ 是行星的矢径扫过面积的速度,应等于椭圆的总面积 πab 除以行星运动周期 T,从而 $h=\frac{2\pi ab}{T}$.而 $p=\frac{b^2}{a}$.从面

$$k=\frac{h^2}{p}=\frac{4\pi a^2b^2}{T^2}\frac{a}{b^2}=\frac{4\pi a^3}{T^2}$$

由开普勒第三定律,$\frac{a^3}{T^2}$ 是常数,因面 k 是与 a,b 无关的常数.这就说明了,太阳对行星的作用力

$$\boldsymbol{F}=m\alpha=-\frac{km}{r^2}\boldsymbol{r}^0$$

指向太阳,其大小与 r^2 成反比,与 m 成正比,其中 k 是由太阳决定的比例常数.这就是万有引力定律.

现在反过来,设万有引力定律成立,推出天体运动的定律.

首先,由天体受力指向太阳,加速度 α 与 $\boldsymbol{r}$ 共线,可知 $h=r^2\dot{\varphi}$ 是常数,也就是天体矢径扫过面积的速度是常量,这就是开普勒第二定律.

由万有引力定律,

$$\alpha=-\frac{k}{r^2}\boldsymbol{r}^0=-ku^2\boldsymbol{r}^0,\quad 即\quad -h^2u^2\left(\frac{\mathrm{d}^2u}{\mathrm{d}\varphi^2}+u\right)\boldsymbol{r}^0=-ku^2\boldsymbol{r}^0$$

于是

$$\frac{\mathrm{d}^2u}{\mathrm{d}\varphi^2}=\frac{k}{h^2}-u$$

两边同乘以 $2\frac{\mathrm{d}u}{\mathrm{d}\varphi}$,再对 φ 求积分,得

$$\left(\frac{\mathrm{d}u}{\mathrm{d}\varphi}\right)^2=C_1+\frac{2k}{h^2}u-u^2=C_2-\left(u-\frac{k}{h^2}\right)^2$$

其中 C_1,C_2 是常数,且显然 $C_2\geqslant 0$.不妨写 $C_2=C^2$,$C\geqslant 0$.

$$\frac{\mathrm{d}u}{\mathrm{d}\varphi}=\pm\sqrt{C^2-\left(u-\frac{k}{h^2}\right)^2}$$

$$\pm\,\mathrm{d}\varphi=\frac{\mathrm{d}u}{\sqrt{C^2-\left(u-\frac{k}{h^2}\right)^2}}$$

$$\mp(\varphi-\varphi_0)=\arccos\left(\frac{u}{C}-\frac{k}{Ch^2}\right)$$

适当选取极轴的方向及 φ 的旋转方向，可设

$$\frac{u}{C}-\frac{k}{Ch^2}=\cos\varphi$$

从而

$$r=\frac{p}{1+e\cos\varphi},\quad 其中\quad e=\frac{Ch^2}{k}\geqslant 0,\quad p=\frac{h^2}{k}>0$$

这是以太阳位置为一个焦点的圆锥曲线方程. 当 $e<1$ 时是椭圆，$e=1$ 时是抛物线，当 $e>1$ 时是双曲线.

当 $e<1$ 时，设椭圆的半长轴和半短轴分别等于 a,b，则 $p=\frac{b^2}{a}$. 又 $h=\frac{2\pi ab}{T}$，故

$$k=\frac{h^2}{p}=\frac{4\pi^2a^2b^2}{T^2}\frac{a}{b^2}=\frac{4\pi^2a^3}{T^2}$$

$$T^2=\frac{4\pi^2}{k}a^3$$

其中 $\frac{4\pi^2}{k}$ 是常数. 这正是开普勒第三定律.

实验九　迭代(一)——方程求解

给定实数域上光滑的实值函数 $f(x)$ 以及初值 x_0，定义数列

$$x_{n+1} = f(x_n), \quad n = 0,1,\cdots \tag{1}$$

$x_n, n = 0,1,\cdots$，称为 $f(x)$ 的一个迭代序列.

函数的迭代是数学研究中的一个非常重要的思想工具. 哪怕是对一个相当简单的函数进行迭代，都可以产生异常复杂的行为，并由此而衍生了一些崭新的学科分支，如分形与混沌. 同时，迭代在各种数值计算算法以及其它学科领域的诸多算法中处于核心的地位. 本书中，我们将通过一系列实验介绍迭代在诸多领域的应用. 首先，我们来探讨利用迭代求解方程(组)的近似解.

9.1　方程求根

给定迭代函数 $f(x)$ 以及一个初值 x_0，利用(1)迭代得到数列 x_n，$n = 0, 1,\cdots$. 如果数列 x_n 收敛于某个 x^*，则有

$$x^* = f(x^*) \tag{2}$$

即 x^* 是方程 $x = f(x)$ 的解. 由此启发我们用如下的方法求方程 $g(x) = 0$ 的近似解.

将方程 $g(x) = 0$ 改写为等价的方程

$$x = f(x) \tag{3}$$

然后选取一初值利用(1)做迭代. 迭代数列 x_n 收敛的极限就是方程 $g(x) = 0$ 的解.

用上述方程求方程的根的一个首要问题是

迭代序列是否收敛？

让我们先从线性函数的迭代开始讨论.

练习 1

设 $f(x) = ax + b$. 利用(1)做迭代得到序列 x_n，$n = 0,1,\cdots$.

(1)写出序列 x_n 的通项公式.

(2)在什么条件下，迭代(1)对任意的初值 x_0 都收敛？

(3)影响收敛性的主要量是什么？它与 $f(x)$ 的一阶导数有什么关系？常数 b 对迭代的收敛性有没有影响？收敛速度的快慢由什么量决定？

(4)对于任意给定的线性方程 $g(x) = Ax + B = 0$，你是否可以将它改写成等价的形式 $x = f(x)$ 使得迭代总是收敛？

对于非线性情况，问题要复杂些．由于非线性方程的解不止一个，迭代序列的极限是否收敛以及收敛到哪一个解不仅与方程有关而且与初值的选取有关．

练习 2

考察用迭代函数 $f(x) = 2\sin(x)$ 求解方程 $g(x) = 2\sin(x) - x = 0$ 的解的情况．

(1)在同一直角坐标系中，画出 $y = f(x)$ 及 $y = x$ 的图象．从图上观察，方程 $x = 2\sin(x)$ 有几个解？

(2)取初值 $x_0 = 0.5$ 做迭代，迭代序列是否收敛？如果收敛，它收敛到哪一个解？取其它初值，观察迭代的结果．是否可以选取到非零的初值 x_0，使得迭代序列收敛到 $g(x) = 0$ 的解 $x = 0$？

(3)你能否解释(2)中观察到的现象？对非线性迭代，迭代序列收敛性与什么因素有关？你能否给出迭代收敛的一个充分的条件？初始值的选取对迭代的收敛性以及收敛到哪一个解有什么影响？

(提示：在一个光滑函数的局部，它可以近似看成一个线性函数．然后，你可以利用线性迭代的有关结论．)．

对于给定的方程 $g(x) = 0$，有无穷多种方式将它改写成等价的形式(3)．但重要的是

如何改写可以使得迭代序列收敛？如何改写才能使得迭代序列收敛得更快？

练习 3

将方程 $x^3 - 2x + 1 = 0$ 改写成各种等价的形式进行迭代，观察迭代是否收敛，并给出解释．

从前面的实验观察中知道，使得迭代序列收敛并尽快收敛到方程 $g(x) = 0$ 的某一解的条件是迭代函数 $f(x)$ 在解的附近的导数的绝对值尽量小．这启

发我们将迭代方程修改成

$$x = h(x) = \lambda f(x) + (1-\lambda)x \tag{4}$$

我们需要选取 λ 使得 $|h'(x)|$ 在解的附近尽量小. 为此,我们可以令

$$h'(x) = \lambda f'(x) + 1 - \lambda = 0$$

得

$$\lambda = \frac{1}{1-f'(x)}$$

于是

$$h(x) = x - \frac{f(x)-x}{f'(x)-1}$$

特别地,如果取 $f(x) = g(x) + x$,则我们得到迭代公式

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)}, \quad n = 0,1,\cdots \tag{5}$$

练习 4

你能否给出上述改进公式的几何解释? 重新做练习 3,使用改进的迭代公式求方程的根. 将它的收敛速度与你得到的其它的迭代公式相比较,哪个更快?

练习 5

选用几种迭代格式求 $\sqrt[3]{2}$ 的近似值,并同改进的迭代格式做比较.

9.2　线性方程组的迭代求解

给定一个 n 元线性方程组

$$\begin{cases} a_{11}x_1 + \cdots + a_{1n}x_n = b_1 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1 + \cdots + a_{nn}x_n = b_n \end{cases} \tag{6}$$

或写成矩阵的形式

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{7}$$

其中 $\boldsymbol{A} = (a_{ij})$ 是 n 阶方阵, $\boldsymbol{x} = (x_1,\cdots,x_n)^{\mathrm{T}}$ 及 $\boldsymbol{b} = (b_1,\cdots,b_n)$ 均为 n 维列向量.

熟知,当矩阵 $\boldsymbol{A}$ 的行列式非零时方程组(7)有唯一解. 如何有效、快速地

寻求大型的线性方程组的数值解是科学工程计算中非常重要的任务. 迭代法常常是求解这些问题的有效方法之一.

用迭代法求解线性方程组的的思想与上节介绍的方程求根的方法是类似的. 假设我们可以将方程组(7)改写成

$$x = Mx + f \tag{8}$$

其中 $M = (m_{ij})$ 是 n 阶矩阵, $f = (f_1, \cdots, f_n)^T$ 是 n 维列向量. 任意给定初始向量 x^0, 由迭代

$$x^{n+1} = Mx^n + f \tag{9}$$

确定向量序列 x^n, $n = 0, 1, \cdots$. 如果 x^n 收敛到向量 x^*, 则有

$$x^* = Mx^* + f$$

即 x^* 是方程组(7)的解.

类似于单变量情形, 我们关心的首要问题是迭代向量列 x^n 是否收敛. 为此, 我们先对 M 是对角阵时进行考察.

练习 6

设 M 为对角阵 $\mathrm{Diag}(m_{11}, \cdots, m_{nn})$. 则向量列 x^n 收敛的条件是什么? 影响收敛性及收敛速度的主要量是什么? 收敛速度与 m_{ii}, $i = 1, \cdots, n$ 有何关系? 常数向量 f 对收敛性有无影响? 初值对收敛性与收敛速度有无影响?

对于较一般的情形, 问题会更加复杂. 但单变量情形的结论以及上面的实验似乎表明, 影响迭代向量列收敛性的主要因素是矩阵 M 的特性. 下面, 我们做进一步的考察.

练习 7

设

$$M = \begin{pmatrix} 0.2 & 0.3 \\ 0.4 & 0.2 \end{pmatrix}$$

(1) 任意取定向量 f 及初始向量 x^0 利用(9)做迭代. 向量列 x^n 是否收敛? 对不同的向量 f 及 x^0 做同样的观察.

(2) 矩阵 M 的特征值是什么? M 是否相似于对角阵?

(3) 你能否给出迭代向量列 x^n 的通项公式? 你能否利用通项公式解释前面观察所得出的结论?

练习 8

分别取

$$M = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$$

$$M = \begin{pmatrix} 0 & 0.5 \\ -0.5 & 1 \end{pmatrix}$$

重新做练习 7 的问题.

练习 9

根据练习 7 与练习 8 的结论做以下问题:

(1)给出向量列 x^n 的通项公式.你能否由此给出迭代(9)收敛的条件?收敛速度由什么决定?

(2)如果矩阵 M 满足 $\max\limits_{1 \leqslant j \leqslant n} \sum\limits_{i=1}^{n} |m_{ij}| < 1$ 或者 $\max\limits_{1 \leqslant i \leqslant n} \sum\limits_{j=1}^{n} |m_{ij}| < 1$,你能否确定迭代(9)一定收敛?

在得到了迭代(9)收敛的条件以后,我们关心的下一个问题是,怎样将方程组(7)改成(8)使得迭代(9)收敛而且尽量收敛得快.

练习 10

给定

$$A = \begin{pmatrix} 3 & 1 \\ 3 & 2 \end{pmatrix}$$

b 任意选取.将方程组(7)改成多种等价的形式(8)做迭代,观察迭代是否收敛?对收敛的迭代格式比较它们收敛速度的快慢.

练习 11

分别取

$$A = \begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix}$$

$$A = \begin{bmatrix} 2 & -1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & -2 \end{bmatrix}$$

回答练习 10 同样的问题.

作为比较,下面我们介绍两种迭代格式.

假设矩阵 A 的对角元素 $a_{ii} \neq 0$, $i = 1,\cdots,n$. 令 $D = \mathrm{Diag}(a_{11},\cdots,a_{nn})$,则我们可以将方程组(7)改写成

$$Dx = (D - A)x + b$$

或

$$x = (I - D^{-1}A)x + D^{-1}b \tag{10}$$

由上式即可确定一种迭代格式.

如果将矩阵 $(I - D^{-1}A)$ 分解为 $U + L$,其中 L , U 分别为下三角阵与上三角阵,则(10)可以进一步该成

$$(I - L)x = Ux + D^{-1}b$$

或

$$x = (I - L)^{-1}Ux + (I - L)^{-1}D^{-1}b \tag{11}$$

上式又可确定另一种迭代格式.

练习 12

用上述两种迭代格式解练习 10 与练习 11 中的方程组,它们是否收敛?收敛速度如何?将它们同你自己找到的迭代格式做比较,谁优谁劣?

练习 13

根据前面的实验结果,你能否找到一种迭代格式,它对任何线性方程组都收敛?对于具有某些特定性质的系数矩阵 A (如 A 正定),你能否找到一种收敛的迭代格式?

9.3 非线性方程组的迭代求解

类似于单变量的方程及线性方程组的求解,用迭代方法可以求更加复杂的非线性方程组的解.

给定非线性方程组

$$\begin{cases} f_1(x_1,\cdots,x_n) = 0 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ f_n(x_1,\cdots,x_n) = 0 \end{cases} \tag{12}$$

将它改写成等价的方程组

$$\begin{cases} x_1 = g_1(x_1,\cdots,x_n) \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ x_n = g_n(x_1,\cdots,x_n) \end{cases}$$

或

$$\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{x}) \tag{13}$$

其中，$\boldsymbol{x}$ 为 n 维列向量 $(x_1,\cdots,x_n)^{\mathrm{T}}$，$\boldsymbol{g} = (g_1(\boldsymbol{x}),\cdots,g_n(\boldsymbol{x}))$ 为 n 维列向量函数. 由上式即确定了一种迭代格式

$$\boldsymbol{x}^{n+1} = \boldsymbol{g}(\boldsymbol{x}^n),\quad n = 0,1,\cdots \tag{14}$$

由于非线性方程组(13)可能有许多解(甚至有无穷多个解)，因此对它的求解比线性方程组的求解要面临更多的挑战. 如如何求出所有的解，如何求出某一特定的解，如何选取恰当的迭代格式，如何保证迭代收敛较快，等等. 我们不打算对此做详细的探讨，仅提出一些问题供读者思考.

问题 1 设 $\boldsymbol{x}^*$ 是方程组(13)的一个解，则 $\boldsymbol{g}(\boldsymbol{x})$ 在 $\boldsymbol{x}^*$ 附近近似线性化的方程组是什么?

问题 2 在什么条件下，只要初值充分靠近(13)的一个解 $\boldsymbol{x}^*$，则迭代(14)一定收敛到 $\boldsymbol{x}^*$?

问题 3 在什么条件下，无论初值如何靠近(13)的解 $\boldsymbol{x}^*$，迭代(14)都不收敛到 $\boldsymbol{x}^*$?

问题 4 你能对迭代的收敛速度说点什么? 迭代的收敛速度与什么有关?

问题 5 给定初值 $\boldsymbol{x}^0$，将 $\boldsymbol{g}(\boldsymbol{x})$ 在 $\boldsymbol{x}^0$ 附近近似线性化，你能否由此给出一个迭代格式，并给出迭代收敛的条件?

问题 6 你能否构造一些其它的迭代格式，并用实例将它同问题 5 中的格式的收敛性及收敛速度做比较?

问题 7 在用迭代法解方程时，什么是线性与非线性之间本质的相似之处与差别?

附录 Mathematica 程序

1. 给定初值 x_0 及迭代函数 $f(x)$，迭代 n 次产生相应的序列

```
Iterate[f_,x0_,n_Integer]:=
    Module[{t={},i,temp=x0},
            AppendTo[t,temp];
        For[i=1,i<=n,i++,temp=f[temp];
            AppendTo[t,temp]];
                t
    ]

f[x_]:=(x+2/x)/2;
Iterate[f,1.,10]
```

2. 由迭代公式(5)产生的迭代序列

```
NTIterate[g_,x0_,n_Integer]:=
    Module[
          {i,var=x0,t={},h},
  h[x_]=Dt[g[x],x];
  For[i=1,i<=n,i++,AppendTo[t,var];
      If[h[var]!=0,var=N[var-g[var]/h[var],20],
          Print["Divided by Zero after",i,
          "'s iterations."];
    Break[]]];
  t
  ]

g[x_]:=x^3-2;
NTIterate[g,1,10]
```

3. 对给定的矩阵 M，数组 f 和初始向量 x^0，由迭代(9)给出的迭代结果

```
LSIterate[m_,f_List,f0_List,n_Integer]:=
  Module[
        {i,var = f0,t = Table[{},{i,n}]},
  For[i = 1,i <= n,i ++,t[[i]] = var;var = m.var + f];
  t
  ]

m = {{0.2,0.3},{0.4,0.2}};f = {1,1};f0 = {0,0};
LSIterate[m,f,f0,20]
```

4. 由迭代(10)产生的迭代向量列

```
JacobIterate[a_,b_List,x0_List,n_Integer]:=
  Module[
    {ad = Length[a],i,j,k,var = var1 = x0},
    For[i = 1,i <= ad,i ++,If[a[[i,i]] == 0,
      Print["a[",i,",",i,"] = 0."];Abort[]]];
    For[i = 1,i <= n,i ++,Print[var];
      For[j = 1,j <= ad,j ++,
        var1[[j]] = N[(b[[j]] - Sum[a[[j,k]] * var[[k]],
          {k,ad}])/a[[j,j]] + var[[j]],20]
      ];
      var = var1;
      ];
      ]

a = {{1,2, - 2},{1,1,1},{2,2,1}};b = {1,1,1};x0 = {0,0,0};
JacobIterate[a,b,x0,10]
```

5. 由迭代(11)产生的迭代向量列

```
SeidelIterate[a_,b_List,x0_List,n_Integer]:=
  Module[
    {ad = Length[a],i,j,k,var = x0},
    For[i = 1,i <= ad,i ++,If[a[[i,i]] == 0,
      Print["a[",i,",",i,"] = 0."];Abort[]]];
    For[i = 1,i <= n,i ++,Print[var];
      For[j = 1,j <= ad,j ++,
        var[[j]] = N[(b[[j]] - Sum[a[[j,k]] * var[[k]],
          {k,ad}])/a[[j,j]] + var[[j]],20]
      ];
    ];
  ]

a={{2, - 1,1},{1,1,1},{1,1, - 2}};b={1,1,1};x0={0,0,0};
SeidelIterate[a,b,x0,10]
```

参考读物

1. Mount Holyoke College. 数学实验室. 白峰杉，蔡大用译. 北京：高等教育出版社；施普林格出版社，1998

2. 徐萃薇. 计算方法引论. 北京：高等教育出版社，1985

实验十　寻　　优

10.1　光的折射定律

光在同一介质中沿直线传播，走的是路程最短的路线. 光在反射时服从反射定律，入射角等于反射角，这也是从一点出发经过反射后到达另一点的最近的路线. 看来，光在选择路线上是很“聪明”的，似乎总是选择路程最短的路线. 但是，光从一种介质进入另一种介质时，却不走直线而走折线，没有沿着路程最短的路线运行. 是不是光变“傻”了呢？如果我们仍然认为光是聪明的而决不会做傻事，就可以设想：假如光在不同介质中的速度不同，那么，在速度小的介质中少走一些路程，在速度大的介质中多走一些路程，走出一条折线，即使总路程略有增加，但总时间仍然可能减少，仍然可能是划算的. 这样看来，走折线不但不傻，反而可能是绝顶聪明的表现：走一条最节省时间的路线！到底走怎样的路线才能使时间最短呢？由物理实验知道，光的折射服从折射定律：入射角的正弦与折射角的正弦之比为常数. 这是否就是最省时间的路线呢？下面就来研究这个问题.

虽然光是在三维空间中发生折射，但可以将这个问题简化为下面的平面上的质点的运动问题：

问题 1　直线 l 将平面区域分成两部分，A，B 分别是这两区域内的两点. 一运动质点在两部分区域内的运动速度分别是 u，v，该质点要从 A 点运动到 B 点，沿怎样的路线运动才能使所花的总时间最短？

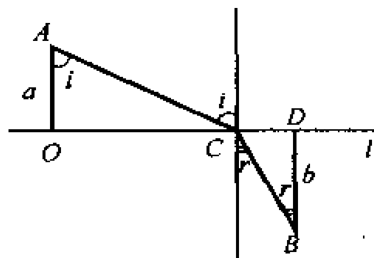


图 10—1

直线 l（代表两种介质的分界面）与 A,B 点的相互位置关系可以刻划如下：作 $AO \perp l$ 于 O，$BD \perp l$ 于 D. 以 O 为原点、OD 为 x 轴正方向、OA 为 y 轴正方向建立直角坐标系. 设 A,B 的坐标分别是 $(0,a),(c,-b)$，则 a,b,c 三个正实数完全刻划了 A,B,l 的相互位置关系.

质点要从 A 运动到 B，必然要经过 l 轴上的某一点 $P(x,0)$. 质点从 A 到 P 显然应走直线段 AP，从 P 到 B 显然应走直线段 PB. 很容易求出 x 与 T 之间的函数关系式：

$$T = \frac{|AP|}{u} + \frac{|PB|}{v}$$

即

$$T = f(x) = \frac{\sqrt{x^2 + a^2}}{u} + \frac{\sqrt{(c-x)^2 + b^2}}{v}$$

这样，问题就归结为求 x 的值 x^* 使函数 $T = f(x)$ 达到最小值. 容易看出，x^* 应在区间 $[0,c]$ 范围内.

实验原理和方法

一般的，设 $y = f(x)$ 是定义在某个区间 $[a,b]$ 上的一元函数，要求最佳点 $x^* \in [a,b]$ 使 $f(x^*)$ 达到该区间内我们所需要的最佳值（最大值或最小值），可以采用以下方法.

1. 均匀搜索法

区间 $[a,b]$ 内有无穷多个值，我们不可能将它们一一穷举，求出最佳点. 但可以将问题离散化，选择一个步长 $d > 0$，一步一步跳着走，依次计算 $x = a, a+d, a+2d, \cdots$ 所对应的 y 值，从中找出最佳的 $y_k = f(a+kd)$，把它对应的 $x_k = a + kd$ 作为最佳点 x^* 的近似值. 则误差 $|x^* - x_k| < d$.

显然，取 d 越小，则精确度越高，但计算量也就越大. 特别在实际应用的情况下，函数关系式 $y = f(x)$ 可能预先不知道，由 x 的每一个值求出对应的 y 值都需要作试验，花费一定的成本，为了节省成本更希望工作量尽量小，希望只试验少数点的 y 值就能得出最佳点. 我们看到，均匀搜索法在离 x^* 还很远的时候也按同样的步长 d，这是在浪费工作量. 应当在一开始选取较大的步长进行比较粗糙的搜索，求出最佳点 x^* 的一个粗糙的近似值. 再在这个近似值附近用较小的步长搜索到更好的近似值. 这样一步步提高精确度，而又能减少工作量的浪费. 因此，有以下方法.

2. 0.618 法

在区间 $[a,b]$ 内插入两个分点

$$x_1 = b - 0.618(b-a) = a + 0.382(b-a), \quad x_2 = a + 0.618(b-a)$$

分别称为区间的 0.382 点和 0.618 点. 也就是说：这两个分点到区间左端点的

距离分别是整个区间长度的 0.382 倍、0.618 倍. 这两个分点将区间分成 3 个小区间. 求出 y 在这两点的值 $y_1 = f(x_1)$, $y_2 = f(x_2)$ 进行比较，求出其中较好的一个. 如果 y_1 比 y_2 好，则可以认为最佳点 x^* 在与 x_1 相邻的两个小区间的范围 $[a, x_2]$ 内，而与 x_1 不相邻的小区间 $(x_2, b]$ 可以去掉不考虑. 剩下的区间 $[a, x_2]$ 的长度缩小为原来长度 $b - a$ 的 0.618 倍. 而且，这个区间 $[a, x_2]$ 内已有一点 x_1 的函数值 $y_1 = f(x_1)$ 是知道的. 并且

$$\frac{x_1 - a}{x_2 - a} = \frac{0.382}{0.618} = 0.618$$

也就是说 x_1 是新区间 $[a, x_2]$ 的 0.618 点.（这是由于我们一开始就选择0.618 使它满足条件 $1 - 0.618 = 0.382 = 0.618^2$，即 0.618 是方程 $1 - x = x^2$ 的正实根.）只要再对该区间的 0.382 点 $x_3 = a + 0.382(x_2 - a)$ 求出对应的函数值 $y_3 = f(x_3)$，将 y_2, y_3 比较优劣，就可以重复前面的步骤将 x^* 所在的区间再次收缩. 类似地，如果在一开始是 y_2 比 y_1 好，则去掉与 x_2 不相邻的小区间 $[a, x_1)$，在 $[x_1, b]$ 中继续搜索 x^*. 此时 x_2 是新区间 $[x_1, b]$ 的0.382点，只要再对这个区间的0.618点 $x_3 = x_1 + 0.618(b - x_1)$ 求对应的函数值 y_3，与 y_2 进行比较，就可以将搜索区间再次缩小.

重复上述过程，每次只多求一个点的函数值，就可将搜索区间缩小到上一次的 0.618 倍. 误差随工作量的增加成指数规律快速减少.

3. 求导法

如果能够求出函数 $y = f(x)$ 对 x 的导函数 $y' = f'(x)$. 求出使 $y' = 0$ 的 $x = x^*$ 值（即 x^* 是方程 $f'(x) = 0$ 的根），或求出使 y' 不存在的 $x = x^*$ 值. 如果函数 f 在 $x = x^*$ 处连续，且当 $x < x^*$ 时 $y' < 0$，当 $x > x^*$ 时 $y' > 0$，则 $x = x_0$ 是极小值点. 类似地可求出极大值点.

如何求出 $f'(x) = 0$ 的根？可以用实验一或实验九所介绍的方程求根的方法. 比如，可先在区间内选一点 x_0 作为 x^* 的最初的近似值，将函数 $y = f(x)$ 在 x_0 附近近似地当作二次函数，用这个二次函数的最小值点 x_1 作为比 x_0 更好的近似值. 这样来不断逼近 x^*. 具体方法如下：

求出 $y_0' = f'(x_0)$. 如果 $y_0' = 0$，则 x_0 可能已是极值点 x^*. 设 $y_0' \neq 0$. 如果能够进一步求出 $f'(x)$ 在 x_0 的导数（即 $f(x)$ 的二阶导数）值 y_0'' 且 $y_0'' \neq 0$，将 $f'(x)$ 在 x_0 附近近似地当作 x 的一次函数，即认为 $f'(x) \approx y_0' + y_0''(x - x_0)$，则 $f'(x) = 0$ 有近似解

$$x_1 = x_0 - \frac{y_0'}{y_0''}$$

用 x_1 代替 x_0，继续此过程直到求出足够满意的极值点.

如果不容易求出 $f(x)$ 的二阶导数值，可用下面的办法将 $f(x)$ 在 $x = x_0$

附近近似地展开成二次函数：先求出 $y_0 = f(x_0), y_0' = f'(x_0)$. 如果 $y_0' \neq 0$ ，则在 x_0 附近另选一点 $x_0 + d$ ，计算 $y_1 = f(x_0 + d)$ 的值.（并且可在 $y_0' < 0$ 时选 $d > 0$ ，在 $y_0' > 0$ 时选 $d < 0$ ，使 $x_0 + d$ 向极小值点 x^* 靠近.）确定二次函数 $g(x) = ax^2 + bx + c$ 的系数 a, b, c 使满足条件

$$g(x_0) = y_0, g'(x_0) = 2ax_0 + b = y_0', g(x_0 + d) = y_1$$

如果 $a > 0$ ，则可将二次函数 $g(x)$ 的最小值点

$$x_1 = -\frac{b}{2a} = x_0 - \frac{y_0'}{2a} = x_0 - \frac{y_0' d^2}{2(y - y_0 - dy_0')}$$

作为最小值点的更好的近似值. 用 x_1 代替 x_0 ，重复刚才的过程. 如果 $a < 0$ ，则二次函数 $g(x)$ 的最小值只能在区间端点取得. 此时可以直接用 $x_0 + d$（或 $x + 2d, x + 3d, \cdots$ 等）代替 x_0 ，重新进行刚才的过程.

可以想到，即使不容易求出 $f(x)$ 的导数，也可以在 x_0 附近将 $f(x)$ 近似地当作二次函数，用这个二次函数的最小值点 x_1 作为 x^* 的更好的近似值. 为此，需要在 x_0 附近另取两个点 $x + d, x + d_1$ ，求出 $x_0, x_0 + d, x_0 + d_1$ 对应的 y 值 y_0, y_1, y_2 ，确定二次函数 $g(x) = ax^2 + bx + c$ 的系数 a, b, c 使 $g(x_0) = y_0$ ， $g(x_0 + d) = y_1$ ， $g(x_0 + d_1) = y_2$.

4. 用数学软件求极值

数学软件很多. 以 Mathematica 为例：

不妨先画出函数 $y = f(x)$ 在区间 $[a, b]$ 上的图象，观察升降变化，找出极值点的大致位置 c 作为初始值. 再在这个初始值附近寻找极值点. Mathematica 语句格式为：

```
Plot[f,{x,a,b}]
FindMinimum[f,{x,c}]
```

下面用以上方法求问题 1 的解，即求函数 $T = f(x)$ 在区间 $[0, c]$ 内的最小值. 并且验证从 A 到 B 最省时间的路线的确满足光的折射定律.

容易看出，当 a, b, c 固定之后，如果将 u, v 同时扩大或者缩小相同的倍数，不会改变最小值点 x^* . 也就是说，x^* 只与 u 与 v 的比值有关. 故在以下总是设 $v = 1$ ，而 u 表示的是两种介质中的速度之比.

练习 1

（1）任意给定 a, b, c, u 的值. 分别用均匀搜索法和 0.618 法求使 T 最小的 x^* 值. 比较这两种方法达到同样的精确度所花费的工作量的大小. 再用数学

软件求出 x^*，与前面两方法求出的 x^* 进行比较.

求出 x^* 的近似值之后，就得到了点 $P(x^*,0)$ 的位置，从而得出了最省时间的路线 APB. 为了验证这条最佳路线是否符合光的折射定律，需要对求出的点 P 计算出入射角 i 与折射角 r 的正弦. 过 P 点作 l 的垂线(即法线) p，则 i,r 分别是 AP,PB 与 p 所夹的锐角.

$$\sin i = \frac{|OP|}{|AP|} = \frac{x^*}{\sqrt{(x^*)^2 + a^2}}, \quad \sin r = \frac{|PD|}{|PB|} = \frac{c - x^*}{\sqrt{(c - x^*)^2 + b^2}}$$

用这两个公式可由 x^* 计算出 $\sin i$ 与 $\sin r$，进而计算出它们的比.

保持 u 不变(即介质不变)，比如取 $u = 1.5$ 或 1.3. 改变 a,b,c，以得到从小到大若干组(比如 8 组) i,r 的值.(为此，可以在固定 a,b 的同时让 c 从小到大取不同的值，再在固定 c 的同时减小 a,b.)对每一组 a,b,c 的值，求出使 T 最小的 x^*，以及相应的 $\sin i$，$\sin r$ 的值以及它们的比值. 将这些数据列表，观察 $\sin i/\sin r$ 是否因 i,r 的改变而改变，并将这个比值与两种介质中的速度比 u 进行比较.

换不同的 u 值，重新做上面的实验，验证：对每一个固定的 u 值，当 a,b,c 变化从而 i,r 变化时，$\sin i/\sin r$ 是否常数？并与 u 值比较.

(2)试用求导法得出 x^* 应满足的条件. 根据这个条件，由给定的 a,b,c,u 值求 x^* 值.

虽然人类很早就发现光从一种介质进人另一种介质时改变方向，发生折射. 人们做了很多的实验，测出了光在折射时的人射角和折射角的大量数据. 但是，当人们试图从这些数据中发现入射角和折射角之间所满足的函数关系时，却遇到了很大的困难. 公元 140 年，希腊天文学家托勒密曾经认为，人射角 i 与折射角 r 成正比，并且用实验方法计算出光在空气、水、玻璃这三种介质中相互进人时 r 与 i 的比值. 但是他所计算出来的这些比值，只对于比较小的入射角才大致相符，而当入射角增大时就不符合了. 比如，下面是一组光从空气进入玻璃时人射角和折射角的数值表.

人射角度数 i	10	20	30	40	50	60	70	80
折射角度数 r	6.7	13.3	19.6	25.2	30.7	35.1	38.6	40.6
i/r	1.49	1.50	1.53	1.59	1.63	1.71	1.81	1.97

由表中数据可以看出，当 i 变化时 i/r 的值并没有保持不变，而是随着 i 的增大而逐渐增大. 那么，到底光的人射角与折射角之间具有怎样的函数关系呢？人类为解决这个问题经历了长达一千多年的时间. 直到 1621 年，荷兰数学

家斯涅尔(W. Snell)才终于找到了人射角与折射角之间的正确的函数关系，这就是：

人射角的正弦与折射角的正弦成正比. 即：$\sin i/\sin r = n$ 是常数.

任何人都很容易由实验数据验证斯涅尔的这个定律的正确性. 比如，读者可以自己对上面的表中的数据验证这一定律，并求出比例常数 n. 但是，斯涅尔怎么会想到要去验证人射角与折射角的正弦的比是否常数呢？这可真不是一件容易的事情，要不然人类怎么会经历了一千多年才想到这一点呢？

但如果换一种思路，猜测光走的是最节省时间的路线. 问题就归结为求函数

$$T = f(x) = \frac{\sqrt{x^2 + a^2}}{u} + \frac{\sqrt{(c - x)^2 + b^2}}{v}$$

的最小值点.

T 对 x 的导函数

$$f'(x) = \frac{x}{u\sqrt{x^2 + a^2}} - \frac{c - x}{v\sqrt{(c - x)^2 + b^2}} = \frac{x}{u|PA|} - \frac{c - x}{v|PB|}$$

$$= \frac{\sin\angle OAP}{u} - \frac{\sin\angle DBP}{v}$$

当 x 在区间 $(-\infty,\infty)$ 内增大时，$\sin\angle OAP$ 单调递增，$\sin\angle DBP$ 单调递减，因面 $f'(x)$ 单调递增.

$$\frac{\mathrm{d}T}{\mathrm{d}x} = 0 \quad 即 \quad \frac{\sin\angle OAP}{u} = \frac{\sin\angle DBP}{v}$$

有唯一的解 $x = x^*$，$0 < x^* < c$. 当 $x < x^*$ 时 $f'(x) < 0$，T 递减；当 $x > x^*$ 时 $f'(x) > 0$，T 递增. 可见当 $x = x^*$ 时 T 最小. 面此时 $P(x^*, 0)$ 点满足的条件

$$f'(x^*) = \frac{\sin\angle OAP}{u} - \frac{\sin\angle DBP}{v} = 0 \quad 即是 \quad \frac{\sin i}{\sin r} = \frac{u}{v} = n$$

其中 i,r 分别是 l 的法线方向与 PA,PB 的夹角. 如果 APB 是光从 A 到 B 的路径，i,r 就分别是人射角和折射角. 这与实验得出的光的折射定律恰好吻合. 这说明光的确是走的最节省时间的路线.

光总是沿着最节省时间的路线传播，这一原理称为费尔马(Fermat)原理.

10.2 奔向最优点

问题 2 设 A_1,A_2,A_3 是平面上的三个不同的点. 在平面上求一点 P，使

它到这三个已知点的距离之和 $s = |PA_1| + |PA_2| + |PA_3|$ 最小.

在平面上建立直角坐标系. 设已知点 A_i 的坐标为 (a_i,b_i) ($i = 1,2,3$). 所求点 P 的坐标为 (x,y) . 则

$$s = \sqrt{(x-a_1)^2+(y-b_1)^2} + \sqrt{(x-a_2)^2+(y-b_2)^2} + \sqrt{(x-a_3)^2+(y-b_3)^2}$$

我们的任务就是要求这个二元函数 $s = f(x,y)$ 的最小值点 $P^*(x^*,y^*)$.

先介绍一种用力学模型来解这个问题的方法:假想将三个已知点的位置 A_1,A_2,A_3 按尺寸画在一个平的薄板上,在 A_1,A_2,A_3 三点各挖一个小洞. 取三根长度相同的细线(其长度 L 大于 A_1,A_2,A_3 两两之间的距离). 将三根线的一端拴在一起作为 P 点的位置,另一端分别从三个洞 A_1,A_2,A_3 中从上到下穿出,分别系在质量相等的三个重物上. 让薄板保持水平,三个重物自由下垂,并假定细线可以在洞中无摩擦地滑动. 现在要问:当三个重物静止不动时,三条线的公共端点 P 的位置满足什么条件?

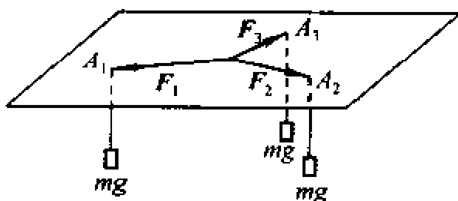


图 10—2

设三个重物的质量都是 m . 首先,由力学原理,系统处于平衡位置时,三个重物的总势能 E 达到最小值. 以平板所在位置的高度为 0. 则第 i 个重物的重心高度 $h_i = |PA_i| - L < 0$,其势能 $E_i = mgh_i$. 三个重物的总势能为

$$\begin{aligned} E &= mgh_1 + mgh_2 + mgh_3 \\ &= (|PA_1| + |PA_2| + |PA_3|)mg - 3mgL \end{aligned}$$

由于 mg 及 L 都是常量, E 最小当且仅当 $s = |PA_1| + |PA_2| + |PA_3|$ 最小. 因此,系统处于平衡位置时, P 的位置 P^* 就是我们所寻找的使 s 最小的点.

另一方面,系统处于平衡位置时, P 点受的合力为 0. 先假设 P 点的平衡位置 P^* 与 A_1,A_2,A_3 三点都不重合. 则 P 点只受到来自三条线的拉力 F_1, F_2,F_3 ,其大小相等,等于每个重物的重量 mg ,其方向分别指向 A_1,A_2,A_3 三点. 三个大小相等的力 F_1,F_2,F_3 平衡的条件为:这三个力两两的夹角都等于 $120°$. 也就是说:线段 PA_1,PA_2,PA_3 两两的夹角等于 $120°$. 事实上,只要三角形 $\triangle A_1A_2A_3$ 的三个内角都小于 $120°$,在三角形内就存在点 P 满足这一条件,从面是所求的最优点.

如果 $\triangle A_1A_2A_3$ 的某个角 A_i 大于或等于 120°，则 P 点在合力作用下被拉到 A_i 点，A_i 就是所求的最小值点.

以上的力学系统是在重力作用下自动达到平衡的. 如果 P 点所处的位置不是最小值点 P^*，则三个拉力 F_1, F_2, F_3 的合力 $\sum F \neq 0$. P 点在 $\sum F$ 的作用下总是向缩小势能 E 从而缩小 s 的方向移动. 事实上，$\sum F$ 的方向是使 E 下降得最快的方向，从而是使 s 下降得最快的方向. P 点在 $\sum F$ 的作用下最快速地奔向 P^* 点.

问题 2 本来是个数学问题，以上却是通过建立一个力学模型使它获得了圆满解决. 其中应用的力学原理是：系统处于平衡位置时势能最小，合力为零. 既然是数学问题，这个力学原理应当可以翻译成数学原理，力学解法应当可以翻译成数学解法. "势能"相当于目标函数 $s = f(x,y)$. $P(x,y)$ 点的"平衡位置"就是使 s 达到最小值的点. 现在要问的是："力"相当于数学上的什么量? "合力"为零相当于数学上的什么条件?

我们从选定的一个初始点 $P_0(x_0,y_0)$ 出发. 假设 P_0 与 A_1, A_2, A_3 都不重合，则在 P_0 点的附近，函数 $s(x,y)$ 可以近似地看作一次函数：

$$s(x,y) \approx s(x_0,y_0) + a(x - x_0) + b(y - y_0)$$

其中 a, b 是与 x, y 无关的常数. 事实上，$a = \dfrac{\partial s}{\partial x}$ 和 $b = \dfrac{\partial s}{\partial y}$ 分别是在 (x_0,y_0) 点 s 对于 x, y 的偏导数. 将 x, y, s 的微小增量 $x - x_0$, $y - y_0$, $s(x,y) - s(x_0,y_0)$ 分别记作 $\Delta s, \Delta x, \Delta y$，则

$$\Delta s \approx \mathrm{d}s = a\Delta x + b\Delta y = (a,b) \cdot (\Delta x, \Delta y)$$

其中 $\mathrm{d}s$ 就是 s 的微分，当 $|P_0P|$ 很小时它可以代表 s 的增量，当 $|P_0P| \to 0$ 时，"相对误差" $(\Delta s - \mathrm{d}s)/|P_0P|$ 也趋于 0. $\mathrm{d}s$ 等于向量 $G = (a,b)$ 与 $\overrightarrow{P_0P} = (\Delta x, \Delta y)$ 的内积. 如果将 G 看作一个"力"，则 $G \cdot \overrightarrow{P_0P}$ 就是在这个"力"的作用下从 P_0 到 P 所作的"功"，它近似地等于"势能" s 的增量. 而在"力" $-G$ 作用下作的功则近似地等于"势能" s 的减少量. 设向量 G 与 $\overrightarrow{P_0P}$ 的方向的夹角为 θ，则 s 在 $\overrightarrow{P_0P}$ 方向上的变化率(即方向导数) $\mathrm{d}s/|P_0P| = |G|\cos\theta$. 梯度 G 的方向(即 θ_0)是 s 增加得最快的方向，这一方向上 s 的变化率(方向导数)就是 G 的大小. 而 $\theta = 180°$ 的方向，也就是 $-G$ 的方向是 s 减少的最快的方向. 让点在"力" $-G$ 的作用下自由运动，则 s 以最快的速度减少，直至 $-G = 0$ 或 G 不存在(本题即 P 与某个 A_i 重合)，则 s 就达到了最小值.

问题 2 中的目标函数 $s = |PA_1| + |PA_2| + |PA_3|$ 是三个函数 $s_i = |PA_i|$ ($i = 1,2,3$)之和，因此 s 的梯度也是这三个函数的梯度 G_i 之和. 对于点 $P_0(x_0,y_0)$，函数 s_i 下降得最快的方向显然是由 P_0 指向 A_i 的方向，而在这

一方向上的方向导数 $\frac{\mathrm{d}s_i}{\mathrm{d}(P_0A_i)} = -1$，可见 $-G_i$ 是 $\overrightarrow{P_0A_i}$ 方向上的单位向量. $-G = -G_1 - G_2 - G_3$ 是三个单位向量 $-G_i$（$i = 1,2,3$）之和（即这三个“力”的合力）. s 最小的条件是 $-G = 0$. 但三个单位向量之和为零的条件就是它们两两的夹角为 120°，这正是我们用力学模型得到的结果.

一般来说，设 n 元函数 $y = f(x_1,\cdots,x_n)$ 在点 $X_0 = (c_1,\cdots,c_n)$ 点附近足够光滑，可以近似地看作 n 元一次函数

$$y = f(c_1,\cdots,c_n) + g_1(x_1 - c_1) + \cdots + g_n(x_n - c_n)$$

则一次项系数 g_i 就是 y 在 X_0 点对 x_i 的偏导数 $\frac{\partial y}{\partial x_i}$，$1 \leqslant i \leqslant n$，向量

$$G = (g_1,\cdots,g_n) = \left(\frac{\partial y}{\partial x_1},\cdots,\frac{\partial y}{\partial x_n}\right)$$

称为这个函数在点 X_0 的梯度，记作 grad$f$. $-G$ 是 y 下降得最快的方向. 当 y 取最小值或最大值时，梯度为 0 或不存在. 我们可以模拟质点在“力” $-G$ 作用下自由运动奔向最小值点的过程：从某个初始点 X_0 出发，沿 $-G$ 方向走一步（移动的距离称为步长）到 P_1；再由 P_1 沿新的 $-G$ 的方向走一步到 P_2 点. 这样沿着 y 下降得最快的方向不断地走下去，直到不能再使 y 减少为止（$G = 0$ 或不存在）. 这个方法称为**最速下降法**.

在最速下降法中怎样选择步长，是一个值得注意的问题. 步长太大，可能走过了最小值点，函数值反而上升. 步长太小，则向最小值点趋近得太慢. 从 X_0 出发沿 $-G$ 方向所到的点可以表示为 $X_0 - \lambda G$，其中正实数 λ 的大小代表了步长的大小（事实上，步长 $= \lambda|G|$）. 我们可以将函数在 $-G$ 方向上看作 λ 的一元函数

$$\varphi(\lambda) = f(X_0 - \lambda G) = f(c_1 - \lambda g_1,\cdots,c_n - \lambda g_n)$$

并且希望进一步看作 λ 的二次函数，求出使它取最小值的 λ^*. 已经知道

$$\varphi(0) = f(X_0),\quad \varphi'(0) = -(g_1^2 + g_2^2 + \cdots + g_n^2)$$

只要再选一个 $\lambda_0 > 0$，求出 $f(\lambda_0)$ 的值，就可以求出一个二次函数 $\psi(\lambda)$ 来近似地代替 $\varphi(\lambda)$，使 $\psi(0) = f(0), \psi'(0) = f'(0), \psi(\lambda_0) = f(\lambda_0)$. 求出这个二次函数的极小值点 λ^*，以

$$X_1 = X_0 - \lambda^* G = (c_1 - \lambda^* g_1,\cdots,c_n - \lambda^* g_n)$$

作为从 X_0 出发走到的下一点. 再由 X_1 出发按同样的方法走到下一点 X_2. 这样一步一步向降低 y 值的地方走去，趋近极小值点.

练习 2

（1）自己选定 $A_i(a_i,b_i)$（$i = 1,2,3$）的坐标. 选择一个初始的点

$P_0(x_0,y_0)$ 作为出发点. 按上面所说的最速下降法逐步趋近最小值点 P^*. 当 P_0 与所有的 A_i 都不重合时,验证 P^*A_i ($i=1,2,3$)两两的夹角是否都等于 120°.

利用 Mathematica 求多元函数极小值的语句

FindMinimum[f,{x,x0},{y,y0}]

求函数 $f(x,y)=\sum_{i=1}^{3}\sqrt{(x-a_i)^2+(y-b_i)^2}$ 在点 $(x0,y0)$ 附近的最小值点和最小值. 注意,语句中的 $x0,y0$ 要输入具体的数值.

(2)在平面上或三维空间选定 4 个或更多个点 A_i ($i=1,2,3,\cdots$),求点 P 的位置使 P 到所有这些点的距离之和最小.

(3)在平面上选定三个点 $A_i(a_i,b_i)$ ($i=1,2,3$),并取定三个正实数 m_i ($i=1,2,3$). 求点 P 的位置,使 $s=m_1|PA_1|+m_2|PA_2|+m_3|PA_3|$ 最小.

10.3 最小二乘法

在进行科学研究和解决实际问题中,经常需要由自变量 x 与因变量 y 的若干组对应数据 (x_i,y_i) ($1\leqslant i\leqslant n$)推测 x 与 y 之间的函数关系 $y=f(x)$. 为此,可以在直角坐标系中画出数据所代表的点. 如果观察发现这些点近似地在一条直线上,就可以认为函数关系是一次函数 $y=ax+b$,剩下的事情是求出一次函数的参数 a,b. 在有的情况下,虽然数据点 (x_i,y_i) 的集合不在一条直线上,但其对数组成的数据点 $(\ln x_i,\ln y_i)$ 的集合也许就在一条直线上,此时 $\ln y$ 与 $\ln x$ 之间成一次函数关系 $\ln y=a\ln x+b$,从而 $y=\lambda x^a$ ($\lambda=e^b$ 是常数). (比如实验十的行星运动的加速度大小 a 与行星与太阳的距离 r 的关系就是这样.)总之,我们经常面临这样的问题:求一条直线 $y=ax+b$ 经过若干已知的数据点 (x_i,y_i). 问题在于,这些已知点虽然近似地在同一条直线上,但并不是精确地在同一条直线上. 即使在理论上 y 应当是 x 的一次函数,由于测量误差和别的偶然因素,(x,y) 的具体数据 (x_i,y_i) 仍然有可能不在一条直线上. 我们不可能找出一条直线完全经过所有这些数据点,只能找一条直线尽可能接近所有这些数据点. 每一个点 (x_i,y_i) 到直线 $y=ax+b$ 的接近程度可以用非负实数 $s_i=(ax_i+b-y_i)^2$ 来衡量,s_i 越小,就说明这个点到这条直线越接近. 于是,直线与所有的数据点总体上的接近程度可以用

$$s(x,b)=\sum_i s_i=\sum_i(ax_i+b-y_i)^2$$

来衡量.我们的任务归结为求 a,b 使 $s(a,b)$ 达到最小值.

更一般的,如果数据点集 (x_i,y_i) 不接近于一条直线,而接近于某一类函数 $y=f(x)$ 的图象(比如 f 是二次多项式、n 次多项式、指数函数、对数函数等),则所选择的 f 应使函数

$$s(f)=\sum_i (f(x_i)-y_i)^2$$

取最小值.用这样的 f 去拟合已知数据点的方法称为**最小二乘法**.

仍考虑用直线 $y=ax+b$ 去拟合数据点的情形.注意 $s(a,b)$ 是 a,b 的二次函数

$$s(a,b)=A_{11}a^2+2A_{12}ab+A_{22}b^2-2A_1a-2A_2b+A_0$$

其中

$$A_{11}=\sum_{i=1}^n x_i^2,\ A_{12}=\sum_{i=1}^n x_i,\ A_{22}=n$$

$$A_1=\sum_{i=1}^n x_iy_i,\ A_2=\sum_{i=1}^n y_i,\ A_0=\sum_{i=1}^n y_i^2$$

而 $s(a,b)$ 对 a,b 的偏导数都是 a,b 的一次多项式:

$$\frac{\partial s}{\partial a}=2(A_{11}a+A_{12}b-A_1),\quad \frac{\partial s}{\partial b}=2(A_{12}a+A_{22}b-A_2)$$

要使 $s(a,b)$ 取最小值,也就是使梯度 $\left(\frac{\partial s}{\partial a},\frac{\partial s}{\partial b}\right)=(0,0)$,满足这一条件的 a,b 就是二元一次方程组

$$\begin{cases}A_{11}a+A_{12}b=A_1\\A_{12}a+A_{22}b=A_2\end{cases}$$

的解.

Mathematica 中有解二元一次方程组解的语句,也有直接求 $s(a,b)$ 在某个初始值附近的最小值点的语句.更直接的是用下面的语句

```
Fit[t,{1,x},x]
```

得到一条直线去拟合数据表 t 中的所有数据点.

练习 3

(1)用下面的第一个语句产生一个数据表 **t**,(其中 **Random[]** 产生区间 [0,1] 的一个随机数),再用第二个语句得到一条直线去拟合它.后面的语句是将数据点和直线画在同一个直角坐标系中,观察它们的接近程度.

```
t = Table[{x + Random[],0.1x + Random[]},{x,0,100}]
t1 = Fit[t,{1,x},x]
p = ListPlot[t]
p1 = Plot[t1,{x,0,100}]
Show[p,p1]
```

(2)用语句 t2=Fit[t,{1,x,x^2},x]求出 x 的二次多项式 t2 去拟合表 t 中的数据，并画出图象，它与直线有多少区别？说明什么问题？

(3)用以下语句求出使函数 $s(a,b)=\sum_{k=1}^{101}(ax_k+b-y_k)^2$ 达到最小值的 a,b. 注意 x_k,y_k 分别是数据表 t 中第 k 对数据的第一个数和第二个数，在下面的语句中分别用 t[[k,1]],t[[k,2]]表示. 由于二次函数 $s(a,b)$ 的最小值点是唯一的，我们可以 (a,b) 的任一组值作为初始值，比如选 (0,0).

```
s[a_,b_]:= Sum[(a*t[[k,1]] + b - t[[k,2]])^2,{k,1,101}]
FindMinimum[s[a,b],{a,0},{b,0}]
```

将求得的最小值点 (a^*,b^*) 与(1)中求得的一次函数的系数相比较，看是否吻合. 在最小值点附近取一些别的点，计算它们对应的 $s(a,b)$ 值，与 $s(a^*,b^*)$ 相比较.

(4)用下面的语句求出 $s(a,b)$ 的多项式展开式 s，以及 s 对 a,b 的偏导数 $g1,g2$. 再求出方程组 $g1=0,g2=0$ 的解，与前面的结果相比较.

```
s = Expand[s[a,b]]
g1 = D[s,a]
g2 = D[s,b]
FindRoot[{g1 == 0,g2 == 0},{a,0},{b,0}]
```

实验十一　最速降线

问题　A,B 是重力场中给定的两点，A 点高于 B 点. 一个在 A 点静止的质点在重力作用下沿着怎样的路线 C 无摩擦地从 A 点滑到 B 点，才能使所花费的时间 T 最短？

使 T 最短的曲线 C 称为**最速降线**.

11.1　时间的计算

不妨以 A 为原点、以向下的方向为 y 轴的正方向、在 AB 与 y 轴所在的平面内建立平面直角坐标系，使 B 点的 x 坐标 a 和 y 坐标 h 都是正实数. 对从 A 到 B 的任一条曲线 $y=f(x)$ ($0\leqslant x\leqslant a$)，我们来计算质点沿此曲线由 A 到 B 所花的时间 T .

由 A 到 B 的最短路径是直线段 $y=\dfrac{h}{a}x$ ($0\leqslant x\leqslant a$). 质点沿直线段 AB 的运动是匀加速运动. 质点在 A,B 两点的速度分别是

$$v_A=0,\quad v_B=\sqrt{2gh}$$

质点在整个直线段 AB 的平均速度 $\bar{v}=(v_A+v_B)/2=\sqrt{2gh}/2$. 总时间

$$T=\frac{|AB|}{\bar{v}}=\sqrt{\frac{2(a^2+h^2)}{gh}}$$

质点沿着曲线 $C:y=f(x)$ $(0\leqslant x\leqslant a)$ 从 A 到 B 所花的时间 $T=T(C)$ 依赖于 C 的选取. 质点在曲线上坐标为 (x,y) 的点 P 的速度为 $v_P=\sqrt{2gy}$. (为了计算方便起见，也可取 $v_P=\sqrt{y}$ ，这样使质点沿每条路线运行的总时间都扩大了 $\sqrt{2g}$ 倍，原来的最速降线仍然是最速降线.)

为了计算总时间 T ，可在 x 的取值区间 $[0,a]$ 内插入 $n-1$ 个分点 $x_k(1\leqslant k\leqslant n-1)$ 使 $0=x_0<x_1<x_2<\cdots<x_{n-1}<x_n=a$. 这样就将 $[0,a]$ 分成 n 个小段 $\Delta_k=[x_{k-1},x_k]$ ，每小段长度 $d_k=x_k-x_{k-1}$. 通常可设 $x_1,x_2,\cdots,x_{n-1}$ 将区间 $[0,a]$ n 等份，即所有的 d_k 相等，等于 a/n ，而 $x_k=ka/n$ 对 $0\leqslant k\leqslant n$ 成立. 但如果曲线用参数方程

$$x=x(u),\quad y=y(u),\quad u\in[u_0,v]$$

给出，并且 x 是 u 的增函数，则可在 u 的变化区间 $[u_0,v]$ 中插入 $n-1$ 个分点

$u_k = u_0 + k(v - u_0)/n$（$1 \leqslant k \leqslant n-1$）将这个区间分成 n 等份，则在 x 变化区间 $[0,a]$ 得到相应的 $n-1$ 个点 $x_k = x(u_k)$（$1 \leqslant k \leqslant n-1$），将 $[0,a]$ 分成 n 小段.

将 x 的变化区间 $[0,a]$ 分成 n 个小段之后，曲线 C 相应地被分成 n 小段 $C_k: y = f(x)$ $(x_{k-1} \leqslant x \leqslant x_k)$，$1 \leqslant k \leqslant n$. 对每个 $0 \leqslant k \leqslant n$，记 $y_k = f(x_k)$，记 A_k 是坐标为 (x_k, y_k) 的点. 注意 $y_0 = 0$ 和 $y_n = h$ 不能改变，$A_0 = A, A_n = B$ 是固定点. 而其余 A_k 及其纵坐标 y_k 随着曲线 C 的不同选取而改变. 曲线 C 被分成的每小段 C_k 的起点和终点分别是 A_{k-1} 和 A_k. 如果 n 比较大，并且每个 d_k 都比较小，则 C_k 可近似地看成从 A_{k-1} 到 A_k 的直线段. 质点在 A_{k-1}, A_k 两点的速度分别是 $\sqrt{2gy_{k-1}}$，$\sqrt{2gy_k}$，在直线段 $A_{k-1}A_k$ 内的平均速度为 $(\sqrt{2gy_{k-1}} + \sqrt{2gy_k})/2$，质点经过这条直线段的时间是

$$\tilde{T}_k = \frac{2\sqrt{d_k^2 + (y_k - y_{k-1})^2}}{\sqrt{2gy_k} + \sqrt{2gy_{k-1}}}$$

总时间 T 近似地等于

$$\tilde{T} = \sum_{k=1}^{n} \tilde{T}_k$$

而准确的总时间

$$T = \lim_{n \to \infty} \tilde{T} = \int_0^a \sqrt{\frac{1 + (y_x')^2}{2gy}} \mathrm{d}x$$

可用求原函数的方法或数值方法（如梯形公式或辛普森公式）求出 T 的值.

如果曲线 C 由参数方程 $x = x(u), y = y(u)$，$u \in [u_0, v]$ 给出，且 $x(u)$ 是区间 $[u_0, v]$ 上的单调递增可微函数，则

$$T = \int_{u_0}^{v} \sqrt{\frac{(x'(u))^2 + (y'(u))^2}{2gy(u)}} \mathrm{d}u$$

练习 1

（1）对从 A 到 B 的以下曲线 C：(i) 直线段，(ii) 圆弧，(iii) 抛物线段，计算时间 T 的近似值. 通过比较，看质点沿着上述哪一种曲线 C 从 A 到 B 所花的时间更少. 注意从 A 到 B 的圆弧和抛物线段都不是唯一的. 你可自己任选一条. 并且可以尝试不同的选择方案，看哪一种更好.

你能否猜出另一种曲线，使时间进一步减少？

（2）直线段 AB 的中点 M 的坐标为 $(a/2, h/2)$. 保持 M 的横坐标不变，将它的纵坐标改变为某个值 $y > h/2$，得到点 $P(a/2, y)$. 分别取曲线 C 为：

(i)折线 APB,(ii)从 A 经过 P 到 B 的抛物线段.计算质点经 C 从 A 到 B 所花的时间 T.

按照实验十所说的方法,选择 y 的适当的值 y^*,使 T 最小.

11.2 寻找最速降线

总时间 T 依赖于曲线 C 的选取.取定一个 n,在区间 $[0,a]$ 中插入 n 等分点 $x_k = ka/n$ $(1 \leqslant k \leqslant n-1)$,从而在曲线上得到相应的 $n-1$ 个分点 $A_k(x_k,y_k)$.则 C 可近似地看作折线 $AA_1A_2\cdots A_{n-1}B$,而 T 可近似地看作各分点的纵坐标 y_k $(1 \leqslant k \leqslant n-1)$的函数

$$T = T(y_1,y_2,\cdots,y_{n-1}) = \sum_{k=1}^{n} \frac{2\sqrt{d^2+(y_k-y_{k-1})^2}}{\sqrt{2gy_k}+\sqrt{2gy_{k-1}}}$$

其中 $d = x_k - x_{k-1} = a/n$.问题就变成求这个 $n-1$ 元函数的最小值.

练习 2

取定一组 a,h 值.用 Mathematica 求多元函数最小值的语句求 $n-1$ 元函数 $T=T(y_1,y_2,\cdots,y_{n-1})$ 的最小值点 $Y^*=(y_1^*,y_2^*,\cdots,y_{n-1}^*)$.为此,首先选定一个 n 值,比如选 $n=16$.利用 Mathematica 语句定义出 **T** 依赖于 15 个变量 **y1,y2,…,y15** 的表达式.还要选定自变量 (**y1,y2,…,y15**) 的一组初始值 (**h1,h2,…,h15**).比如,可以取从 A 到 B 的直线段作为初始曲线,以直线 AB 上对应点的坐标 $kh/16$ $(1 \leqslant k \leqslant 15)$依次作为初始值 **h1,h2,…,h15**.然后运行语句

```
FindMinimum[T,{y1,h1},{y2,h2},…,{y15,h15}]
```

求得最小值点 $Y^*=(y_1^*,y_2^*,\cdots,y_{15}^*)$.

依次将点 $A(0,0)$,$A_1(d,y_1^*)$,$A_2(2d,y_2^*)$,…,$A_{15}(15d,y_{15}^*)$,$B(nd,h)$ 连成折线或光滑曲线,就得到从 A 到 B 的最速降线的近似形状.

取不同的 a,h 值,观察所得的结果.特别取一组比值 h/a 很小的 a,h,观察最速降线先下降后上升的情形.

如果不用 Mathematica,可用别的语言编程按前一个实验中介绍的最速

下降法(沿负梯度的方向趋向最小值点)求最小值点.根据本问题的特点,相邻变量取值应当很接近,可考虑采用下面的对各变量轮流优化的方法.

仍以 $n = 16$ 为例.

以直线段 AB 作为最速降线的初始的近似解 C_0 ,逐步改进,寻找更好的近似解.

取点 $A_8(a/2,y_8)$,其横坐标 $a/2$ 为 A,B 横坐标的平均值,纵坐标 y_8 可以变动,则质点沿折线 AA_8B 运行的时间 T 是 y_8 的一元函数.可用求一元函数极值点的方法求 y_8 的最好值使质点沿折线 $C_1:AA_8B$ 运行的时间最少. C_1 是比 C_0 更好的最速降线近似解.

固定 A,A_8,B 点.在 A,A_8 之间取点 $A_4(a/4,y_4)$,求 y_4 的最好值 h_4 使质点沿折线 AA_4A_8 运行的时间最少.在 A_8 与 B 之间取点 $A_{12}(3a/4,y_{12})$,求 y_{12} 的最好值使质点沿折线 $C_2:A_8A_{12}B$ 运行的时间最少.折线 $C_2:AA_4A_8A_{12}B$ 是比 C_1 更好的解.

重复刚才的过程:对组成折线 C_2 的每一条直线段 $A_{4k-4}A_{4k}$ ($k = 1,2,3,4$) (约定 $A_0 = A,A_{16} = B$), 在它的两个端点之间插入一点 $A_{4k-2}((4k-2)a/16,y_{4k-2})$,选择 y_{4k-2} 的最好值使质点沿折线 $A_{4k-4}A_{4k-2}A_{4k}$ 运行的时间最少,用这条折线代替直线段 $A_{4k-4}A_{4k}$,就得到从 A 依次经过点 A_{2k} ($1 \leqslant k \leqslant 7$)到 B 点的折线 C_3 来代替 C_2 ,它是比 C_2 更好的解.然后再在构成 C_3 的每一条直线段 $A_{2k-2}A_{2k}$ ($1 \leqslant k \leqslant 8$)的两个端点之间插入一点 $A_{2k-1}((2k-1)a/16,y_{2k-1})$,选择其纵坐标 y_{2k-1} 使质点沿折线 $A_{2k-2}A_{2k-1}A_{2k}$ 运行的时间最少,用这条折线代替直线段 $A_{2k-2}A_{2k}$,这样得到从 A 依次经过点 A_k ($1 \leqslant k \leqslant 15$)再到 B 的新的折线 C_4 来代替 C_3 ,它比 C_3 更好.

如果还希望增加精度,当然还可以在构成折线 C_4 的每一条直线段的两个端点之间再插入新的点.假如不再插入新的点,则在 A,B 之间插入的 15 个点的位置仍有调整的余地.注意奇数点 $A_1,A_3,\cdots,A_{15}$ 的纵坐标是在固定偶数点 $A_2,A_4,\cdots,A_{14}$ 的情况下的最好值.现在反过来面定奇数点 $A_1,A_3,\cdots,A_{15}$ 的位置,重新调整偶数点 $A_2,A_4,\cdots,A_{14}$ 的纵坐标 $y_2,y_4,\cdots,y_{14}$ 的值分别使质点经过折线 $A_1A_2A_3,A_3A_4A_5,\cdots,A_{13}A_{14}A_{15}$ 的时间最少.再固定偶数点位置而调整所有的奇数点纵坐标.重复这个过程,轮流调整奇数点和偶数点的纵坐标,每调整一次都使质点经过所得的折线的时间端短一些,得到最速降线的更好的解.直到在所需的精度下时间不再能缩短为止,就得到了在插入 15 个点的情况下的最优解.

以上方法还可作这样的改进:每次在两个已有的点 A_i,A_j 之间插入点 A_k 之后,用从 A_i 经过点 A_k 到 A_j 的抛物线段来代替原先从 A_i 到 A_j 的曲线段,并选择 A_k 的纵坐标 y_k 的值使质点经过这条抛物线段的时间最短.这样,最后得

到的就是由抛物线段组成的曲线作为最速降线的近似解.

11.3 最速降线的形状

前面用求多元函数极值的数值方法求得了最速降线的近似形状.但我们希望知道:精确的最速降线到底是什么曲线? 它的曲线方程是什么?

在前一个实验中,我们知道光总是走最省时间的路线.由此想到,可以让质点模仿光的行为,按照光的折射定律运行,这样走出的就应当是最速降线.

先将这个问题离散化,取一个很小的正实数 d ,若干个不同高度 $h_k = kd$ ($k = 0,1,2,\cdots$),用一组平行于 x 轴的直线 $y = h_k$ ($k = 0,1,2,\cdots$)将质点所要通过的空间分成很多层,相邻平行线 $y = h_{k-1}$,$y = h_k$ ($k = 1,2,3,\cdots$)之间所夹的部分是第 k 层.当 d 很小的时候,可以认为质点在每一层内部的速度不变,以第 k 层内的平均速度 $v_k = \dfrac{\sqrt{2gh_{k-1}} + \sqrt{2gh_k}}{2}$ 作为质点在这一层中的速度.质点从第 k 层进入第 $k+1$ 层时速度由 v_k 变为 v_{k+1} ,发生折射,满足折射定律

$$\frac{\sin \alpha_k}{\sin \alpha_{k+1}} = \frac{v_k}{v_{k+1}} \quad 即 \quad \frac{\sin \alpha_k}{v_k} = \frac{\sin \alpha_{k+1}}{v_{k+1}} \tag{1}$$

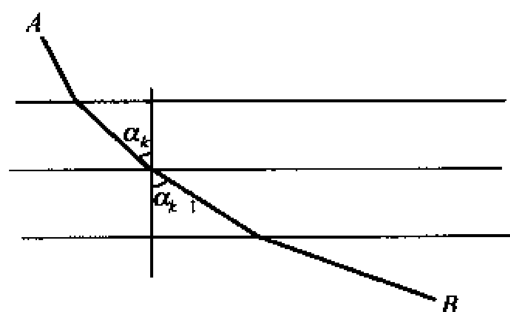


图 11－1

其中入射角 α_k 与折射角 α_{k+1} 分别是质点在第 k 层、第 $k+1$ 层中的运行路线与 y 轴方向所夹的锐角.于是由(1)知

$$\frac{\sin \alpha_k}{v_k} = C \tag{2}$$

是一个与 k 无关的常数.令 d 趋于 0,则折线趋于最速降线 C . α_k 成为曲线上一点 $P(x,y)$ 处的切线方向与 y 轴的夹角 α , v_k 是质点在该点的速度 v ,而上述关于 α_k,v_k 的关系式(2)变为关于 α,v 的关系式

$$\frac{\sin\alpha}{v} = C \tag{3}$$

我们有 $v = \sqrt{2gy}$. 再来计算 $\sin\alpha$. 光所走的曲线 C 在点 $P(x,y)$ 的切线方向与 x 轴的夹角为 $\frac{\pi}{2} - \alpha$. 设 C 的方程为 $y = f(x)$ ，则 $\cot\alpha = \tan\left(\frac{\pi}{2} - \alpha\right) = y'_x$ 是函数 $y = f(x)$ 在该点的导数值. 于是 $\sin\alpha = \frac{1}{\sqrt{1+(y'_x)^2}}$. 这样，(3)式就成为 C 所应满足的微分方程

$$\frac{1}{\sqrt{2gy(1+(y'_x)^2)}} = C, \quad \text{或化为} \quad y(1+(y'_x)^2) = C_1$$

其中 C_1 是常数.

引入参数 t 使 $y'_x = \cot t$ ，就得到

$$y = \frac{C_1}{1+\cot^2 t} = C_1 \sin t^2 = \frac{C_1}{2}(1-\cos 2t)$$

$$\mathrm{d}x = \frac{\mathrm{d}y}{y'_x} = \frac{C_1 \sin 2t \mathrm{d}t}{\cot t} = 2C_1 \sin^2 t = C_1(1-\cos 2t)\mathrm{d}t$$

$$x = C_1(t - \frac{1}{2}\sin 2t) + C_2 = \frac{C_1}{2}(2t - \sin 2t) + C_2$$

令 $\theta = 2t$, $R = \frac{C_1}{2}$ ，并考虑到 $x = 0$ 时 $y = 0$ ，故 $C_2 = 0$. 于是曲线 C 的参数方程是

$$x = R(\theta - \sin\theta), \quad y = R(1-\cos\theta)$$

这是半径为 R 的圆沿着 x 轴作无滑动的滚动时，圆周上一固定点画出的曲线，称为**旋轮线**. (请自己验证这个结论.)

按上述参数方程作出的旋轮线总是经过原点 $A(0,0)$. 为了让它也经过 $B(a,h)$ ，需要解方程组

$$\begin{cases} R(\theta - \sin\theta) = a \\ R(1-\cos\theta) = h \end{cases}$$

求出适当的 R,θ . 两式相除消去 R ，再整理得 θ 的一元方程

$$h\theta - h\sin\theta + a\cos\theta - a = 0$$

可用 Mathematica 求这个方程在 $(0,2\pi)$ 范围内的解. 为此，可先用画图语句画出上面的方程左边的函数 $f(\theta)$ 的图像，观察图像与横轴的交点的横坐标的大致的值 x0. 再用求方程根的语句求出在初始值 x0 附近的根的更精确的值 α. 再由

$$R = \frac{h}{1-\cos\alpha}$$

求出 R 的值.

练习 3

选定一组 a,h 值. 按上述方法求出 R,α 的值. 画出旋轮线 A 到 B 的部分

$$x = R(\theta - \sin\theta),\quad y = R(1 - \cos\theta),\quad 0 \leqslant \theta \leqslant \alpha$$

在同一坐标系中画出用练习 2 的方法求出的最速降线的近似形状，观察它们与旋轮线是否吻合. 为了便于辨认，不妨用红色画旋轮线，黑色或绿色画你求出的近似的最速降线.

用练习 1 的方法计算出质点经过旋轮线从 A 到 B 所花的时间 T. 与练习 1 对其它曲线的计算结果相比较，看它是否最小.

旋轮线上从 A 到 B 的时间 T 很容易由积分算出. 注意旋轮线上从参数值 θ 到 $\theta + \mathrm{d}\theta$ 的一段弧长为

$$\begin{aligned}\mathrm{d}s &= \sqrt{\mathrm{d}^2x + \mathrm{d}^2y} = \sqrt{(x'_\theta)^2 + (y'_\theta)^2}\mathrm{d}\theta \\ &= \sqrt{R^2(1-\cos\theta)^2 + R^2\sin^2\theta}\mathrm{d}\theta \\ &= R\sqrt{2(1-\cos\theta)}\mathrm{d}\theta\end{aligned}$$

故

$$T = \int_0^\alpha \frac{\mathrm{d}s}{\sqrt{2gy}} = \int_0^\alpha \frac{R\sqrt{2(1-\cos\theta)}}{\sqrt{2gR(1-\cos\theta)}}\mathrm{d}\theta = \sqrt{\frac{R}{g}}\int_0^\alpha \mathrm{d}\theta = \alpha\sqrt{\frac{R}{g}}$$

用这个公式重新计算 T，与练习 3 算出的结果相比较.

练习 4

利用与求最速降线类似的方法解决下面的问题：在一条直线 l 的同一侧有两个已知点 A,B. 试找出一条从 A 到 B 的曲线，使这条曲线绕 l 旋转所得的曲面的面积最小.

你能否通过观察所得曲线的形状猜出它是什么曲线?

11.4 等时曲线

从一个定点 E 点悬挂一根长为 L 的无重量不可伸缩的细线，细线的另一端悬挂一个质点作成一个单摆. 伽利略通过实验发现，摆动周期与振幅无关.

注意单摆的质点运动的轨迹是以 E 为圆心、L 为半径的圆弧. 设圆弧的最低点为 D,某一方向的最高点为 A. 则质点在重力作用下从静止开始沿圆弧无摩擦地从 P 点滑到 D 点所花的时间 T 就是单摆周期的 1/4. 按照伽利略的实验,T 应当与 P 点的位置无关,而只与 L 有关. 这个现象称为单摆的等时性. 并且有公式

$$\text{摆动周期} \quad 4T = 2\pi\sqrt{\frac{L}{g}}$$

圆弧 AD 上的任意一点 P 的位置可用 $\theta = \angle DEP$ 来表示. 特别 $\alpha = \angle DEA$ 的大小代表单摆的振幅的大小. 点 P 与 A 的高度差为 $L(\cos\theta - \cos\alpha)$,因此质点在 P 点的速度为 $v = \sqrt{2gL(\cos\theta - \cos\alpha)}$. 质点从 P 运行一小段弧到 P_1 点,设这一小段弧所对的圆心角 $\angle PEP_1$ 为 $\delta\theta$. 则弧长为 $L\delta\theta$,质点通过这一小段的时间

$$\Delta t = \frac{L\delta\theta}{\sqrt{2gL(\cos\theta - \cos\alpha)}}$$

于是从 A 到 D 的总时间为

$$T = -\int_\alpha^0 \frac{L\mathrm{d}\theta}{\sqrt{2gL(\cos\theta - \cos\alpha)}} = \int_0^\alpha \frac{L\mathrm{d}\theta}{\sqrt{2gL(\cos\theta - \cos\alpha)}}$$

如果 α 很小,当然 θ 更小,此时

$$\cos\theta - \cos\alpha \approx \left(1 - \frac{1}{2}\theta^2\right) - \left(1 - \frac{1}{2}\alpha^2\right) = \frac{1}{2}(\alpha^2 - \theta^2)$$

$$4T \approx 4\sqrt{\frac{L}{g}}\int_0^\alpha \frac{\mathrm{d}\theta}{\sqrt{\alpha^2 - \theta^2}} = 2\pi\sqrt{\frac{L}{g}}$$

周期 $4T$ 近似地与 α 无关. 但当 α 不太小时,这个计算误差就可能比较大了.

练习 5

对 $\alpha = 5°, 10°, \cdots, 90°$,用数值积分方法(比如梯形法或辛普森法,见实验二)计算

$$\lambda = \frac{T}{\sqrt{L/g}} = \int_0^\alpha \frac{\mathrm{d}\theta}{\sqrt{2(\cos\theta - \cos\alpha)}}$$

列出 λ 与 α 值的对照表. 观察 λ 随 α 的变化而变化的情况. 由此知道质点沿着圆弧作摆动的等时性只是近似成立.

质点沿着什么样的曲线作摆动才具有真正的等时性呢?

练习 6

建立直角坐标系使 y 轴的正方向向下. 任选 R，按参数方程

$$x = R(\theta - \sin\theta),\quad y = R(1 - \cos\theta)$$

作旋轮线. $\theta = \pi$ 时得到的点 $D(\pi R, 2R)$ 是一个最低点，它与最高点（比如原点 $O(0,0)$）的高度差为 $2R$. 任取 $0 \leqslant \alpha < \pi$ 得到旋轮线在 O 与 D 之间的一点 $A(x(\alpha), y(\alpha))$. 设质点从 A 点静止出发沿旋轮线无摩擦地滑到 D 点，用练习 2 所说的方法计算所花费的时间 T. 让质点到了 D 之后继续沿旋轮线自由运动. 则质点将以 D 为平衡位置作摆动，周期为 $4T$.

对不同的出发点 A（也就是不同的 α）计算时间 T，看 T 是否随 α 的改变而改变.

通过以上实验，可以发现旋轮线是真正的等时曲线. 实际上，这一结论也可通过积分运算直接推导出来：

将旋轮线上由参数 θ 决定的点 $(x(\theta), y(\theta))$ 记作 $P(\theta)$. 用积分计算质点从点 $A = P(\alpha)$ 静止开始沿旋轮线无摩擦地滑到最低点 $D = P(\pi)$ 所需的时间 T. 质点在 A 与 D 之间任一点 $P(\theta)$ 的速度

$$v(\theta) = \sqrt{2g(y(\theta) - y(\alpha))} = \sqrt{2gR(\cos\alpha - \cos\theta)}$$

而从点 $P(\theta)$ 到 $P(\theta + \mathrm{d}\theta)$ 的弧长元

$$\begin{aligned}\mathrm{d}s &= \sqrt{(\mathrm{d}x)^2 + (\mathrm{d}y)^2} = \sqrt{(x'(\theta))^2 + (y'(\theta))^2}\mathrm{d}\theta \\ &= R\sqrt{(1 - \cos\theta)^2 + \sin^2\theta}\,\mathrm{d}\theta = R\sqrt{2(1 - \cos\theta)}\,\mathrm{d}\theta\end{aligned}$$

于是

$$\begin{aligned}T &= \int_\alpha^\pi \frac{\mathrm{d}s}{v(\theta)} = \sqrt{\frac{R}{g}}\int_\alpha^\pi \sqrt{\frac{1 - \cos\theta}{\cos\alpha - \cos\theta}}\mathrm{d}\theta \\ &= \sqrt{\frac{R}{g}}\int_\alpha^\pi \sqrt{\frac{2\sin^2\dfrac{\theta}{2}}{2\cos^2\dfrac{\alpha}{2} - 2\cos^2\dfrac{\theta}{2}}}\mathrm{d}\theta \\ &= \sqrt{\frac{R}{g}}\int_\pi^\alpha \frac{2\mathrm{d}\left(\cos\dfrac{\theta}{2}\right)}{\sqrt{\cos^2\dfrac{\alpha}{2} - \cos^2\dfrac{\theta}{2}}} \\ &= \sqrt{\frac{R}{g}}\left[2\arcsin\left(\frac{\cos\dfrac{\alpha}{2}}{\cos\dfrac{\theta}{2}}\right)\right]\Bigg|_{\theta=\pi}^{\alpha} = \pi\sqrt{\frac{R}{g}}\end{aligned}$$

(当 $\theta=\alpha$ 时被积函数分母为 0，T 是广义积分. 不妨用 Mathematica 语句 **NIntegrate** 计算当 $\varepsilon=10^{-n}$（$n=10,11,\cdots,15$）时积分 $I(\varepsilon)=\int_{\alpha+\varepsilon}^{\pi}\sqrt{\frac{1-\cos\theta}{\cos\alpha-\cos\theta}}\mathrm{d}\theta$ 的值，观察 $I(\varepsilon)$ 的变化趋势.)

能否也在一个定点 E 悬挂一条细线、细线另一端固定一个质点，作成一个单摆，使它在摆动时质点沿旋轮线运动呢？可以这样做：以 E 为原点、向下的方向作为 y 轴的正方向建立直角坐标系. 从原点向 x 轴的两个方向用硬质材料各做出旋轮线半个拱(如图 11—2)：

$$x=R(\theta-\sin\theta),\quad y=R(1-\cos\theta),\quad -\pi\leqslant x\leqslant\pi$$

则每半个拱的弧长为

$$\begin{aligned}\int_0^{\pi}\sqrt{(\mathrm{d}x)^2+(\mathrm{d}y)^2}&=\int_0^{\pi}\sqrt{x'(\theta)^2+y'(\theta)^2}\mathrm{d}\theta\\&=R\int_0^{\pi}\sqrt{(1-\cos\theta)^2+\sin^2\theta}\mathrm{d}\theta=4R\end{aligned}$$

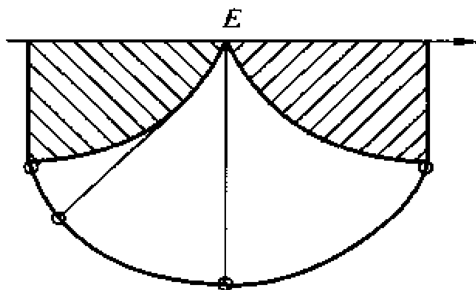


图 11—2

从原点 E 悬挂一根长为 $4R$ 的细线，细线另一端固定一个质点作成一个单摆. 在这个单摆的摆动过程中，由于受到悬挂点 E 两方的硬质旋轮线拱的限制，细线上方接近 E 的部分不能拉成直线，只能绕在旋轮线拱上. 因而质点的轨迹不是圆弧，而是另一个旋轮线. 有兴趣的读者可以自己通过理论推导或设计一个数值实验来验证这一结论. 这样，我们做出来的这个单摆的摆动周期就真正与振幅无关了. 如果振幅很小，则细线受旋轮线拱的影响较小，质点的轨迹与圆弧很接近，这说明了普通的单摆为什么具有近似的等时性.

用旋轮线可以做出具有真正的等时性的单摆，所以，旋轮线又称为**摆线**.

附录　用变分法求最速降线的方程

设 $C: y = y(x)$ ($x \in [0,a]$)是从 $A(0,0)$ 到 $B(a,h)$ 的连续光滑曲线. 则质点沿曲线 C 运行的时间为

$$T = \int_0^a F(y, y')\mathrm{d}x$$

其中

$$F(y, y') = \sqrt{\frac{1 + (y')^2}{2gy}}$$

是 y, y' 的函数. T 不是有限个变量的函数,而是由一个函数 $y = y(x)$ ($x \in [0,a]$)决定,称为**泛函**,可记为 $T(y(x))$. 我们的任务是选择恰当的 $y(x)$ 使 T 取最小值. 这样的求泛函的极值的方法称为**变分法**.

假设已选 $y = y_0(x)$ 使 $T(y_0(x))$ 达到最小值. 则对任意一个其它的函数 $y = y(x)$,有 $T(y(x)) \geqslant T(y_0(x))$. 记 $\delta y = y(x) - y_0(x)$,则 δy 也是 $[0,a]$ 上连续可导函数,且 $(\delta y)(0) = (\delta y)(a) = 0$. 设 λ 是绝对值很小的实数, $y = y(x,\lambda) = y_0(x) + \lambda\delta y$ 也代表一条从 A 到 B 的连续光滑曲线,且 $T(y(x,\lambda)) \geqslant T(y_0(x))$. $T(y(x,\lambda))$ 可以看作 λ 的一元函数,记作 $\varphi(\lambda)$, 且当 $\lambda = 0$ 时达到最小值. 因此, $T = \varphi(\lambda)$ 对 λ 的导数在 $\lambda = 0$ 处为 0. 而

$$\begin{aligned}
\varphi'(\lambda) &= \int_0^a \frac{\partial F(y(x,\lambda), y'_x(x,\lambda))}{\partial \lambda}\mathrm{d}x \\
&= \int_0^a \left[F_y \frac{\partial y(x,\lambda)}{\partial \lambda} + F_{y'} \frac{\partial y'_x(x,\lambda)}{\partial \lambda}\right]\mathrm{d}x \\
&= \int_0^a [F_y\delta y + F_{y'}\delta y']\mathrm{d}x
\end{aligned}$$

其中 $F_y = \dfrac{\partial F(y(x,\lambda), y'(x,\lambda))}{\partial y}$, $F_{y'} = \dfrac{\partial F(y(x,\lambda), y'(x,\lambda))}{\partial y'}$, $\delta y' = \dfrac{\mathrm{d}\,\delta y}{\mathrm{d}\,x}$. 由分部积分公式得

$$\int_0^a F_{y'}\delta y'\mathrm{d}x = (F_{y'}\delta y)\big|_0^a - \int_0^a \delta y \frac{\mathrm{d}\,F_{y'}}{\mathrm{d}\,x}\mathrm{d}x = -\int_0^a \delta y \frac{\mathrm{d}\,F_{y'}}{\mathrm{d}\,x}\mathrm{d}x$$

于是

$$\varphi'(0) = \int_0^a \left(F_y - \frac{\mathrm{d}}{\mathrm{d}x}F_{y'}\right)\delta y\mathrm{d}x = 0 \tag{1}$$

注意,若一个连续函数 $f(x)$ 在某个区间 $[x_1, x_2]$ 内不恒为 0,设 $f(x_0) \neq 0$,则存在正实数 ε 使 $f(x)$ 在区间 $[x_0 - \varepsilon, x_0 + \varepsilon]$ 内与 $f(x_0)$ 同号. 又可选取连续可导函数 δy 在区间 $[x_0 - \varepsilon, x_0 + \varepsilon]$ 内恒为正,而在此区间外恒为 0. 这

将导致 $\int_{x_1}^{x_2} f(x)\delta y \mathrm{d}t$ 与 $f(x_0)$ 同号而不为 0. 这说明了：如果 $\int_{x_1}^{x_2} f(x)\delta y \mathrm{d}t = 0$ 对任意连续可导函数 δy 成立，则 $f(x)$ 在区间 $[x_1, x_2]$ 内恒等于 0.

于是由(1)式得

$$F_y - \frac{\mathrm{d}}{\mathrm{d}x} F_{y'} = 0 \tag{2}$$

这就是 $y = y_0(x)$ 满足的微分方程，称为**欧拉方程**. 在我们的情况下，F 只依赖于 y, y' 而不含 x，欧拉方程可展开为

$$F_y - F_{y'y} y' - F_{y'y'} y'' = 0$$

两边逐项乘以 y'，则可验证左边成为完全微分

$$\frac{\mathrm{d}}{\mathrm{d}x}(F - F_{y'} y') = 0$$

从而

$$F - F_{y'} y' = C \tag{3}$$

C 是常数. 在最速降线问题中，$F = \sqrt{\frac{1 + (y')^2}{2gy}}$. 于是(4)式成为

$$\sqrt{\frac{1 + (y')^2}{2gy}} - \sqrt{\frac{(y')^2}{2gy(1 + (y')^2)}} = C$$

再化简后成为

$$\frac{1}{\sqrt{2gy(1 + (y')^2)}} = C, \quad 或 \quad y(1 + (y'_x)^2) = C_1$$

其中 C_1 是另一常数. 这正是我们在第 3 节中用光的折射定律得出的结论. 在第 3 节已经由它得出了最速降线满足的方程，即旋轮线的参数方程.

实验十二　迭代(二)——分形

在我们生活着的大千世界里，除了有像房屋建筑、公路桥梁、汽车、飞机、轮船以及各种劳动生活工具等这些人造的形态规则的几何形体之外，更广泛地充满了诸如花草树木、山川河流、烟雾云彩等形态极不规则的几何形体. 大自然在向人们展示其美丽多变形态的同时，也提出了难以回答的询问：如何描述复杂的自然表象？如何分析其内在的机理？科学家与艺术家一直在苦苦追寻着这些问题的答案，并力图从传统的欧几里得几何体系中解放出来. 最近几十年，一些科学家开始朦胧地"感觉"到了另一个几何世界的存在，这个几何世界的描述对象是自然界的几何形态. 七十年代，美国科学家 B. Mandelbrot 用 Fractal(原意是碎片、分数等)这个词来定义这门新的几何学科——分形几何学. 分形几何把自然形态看作是具有无限嵌套层次的精细结构，并且在不同尺度下保持某种相似的属性，于是在简单的迭代过程中就可以得到描述复杂的自然形态的有效方法.

尽管分形的提出只有二十多年的时间，但它已经在自然科学的诸多领域如数学、物理、化学、材料科学、生命科学、地质、地理、天文、计算机乃至经济、社会、艺术等极其广泛的领域有着重大的应用. 可以毫不夸张地说，"分形是大自然的几何学"，"分形处处可见".

本实验的目的是以迭代的观点介绍分形的基本特性以及生成分形图形的基本方法，使读者在欣赏美丽的分形图案的同时对分形几何这门学科有一个直观的了解，并从哲理的高度理解这门学科诞生的必然，激发读者探寻科学真理的兴趣.

12.1　生成元

早在上世纪末及本世纪初，一些数学家就构造出一些边界形状极不光滑的图形. 由于这类图形长期以来被视为"不可名状的"或"病态的"，因而，只有当人们需要反例时才想到它们. 这类图形的构造方式都有一个共同的特点，即最终图形 F 都是按照一定的规则 R 通过对初始图形 F_0 不断修改得到的. 其中最具有代表性的图形是 Koch 曲线，Koch 曲线的构造方式是：

给定一条直线段 F_0，将该直线三等分，并将中间的一段用以该线段为边

的等边三角形的另外两条边替代，得到图形 F_1（如图 12—1 所示）. 然后，再对图形 F_1 中的每一小段都按上述方式修改，以至无穷. 则最后得到的极限曲线 $F=\lim\limits_{k\to\infty}F_k$ 即是所谓的 Koch 曲线.



图 12—1

Koch 曲线的修改规则 R 是将每一条直线段 F_0 用一条折线 F_1 替代，我们称 F_1 为该分形的生成元. 分形的基本特性完全由生成元决定. 因此，给定一个生成元，我们就可以生成各种各样的分形图形. 以下是几个经典的分形图形及其生成元：

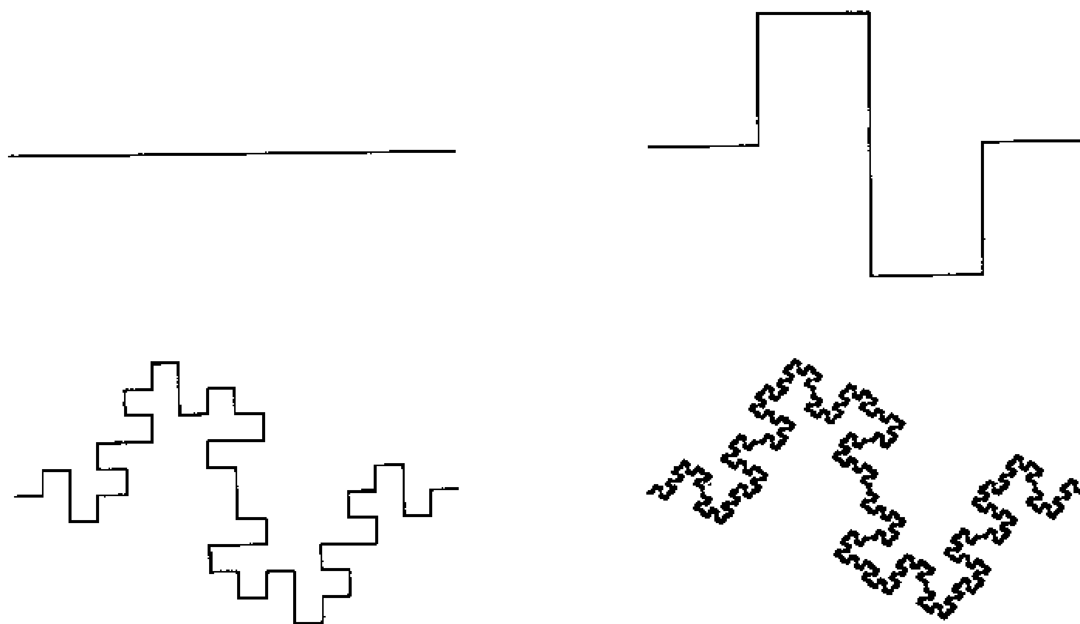


图 12—2 Minkowski“香肠”

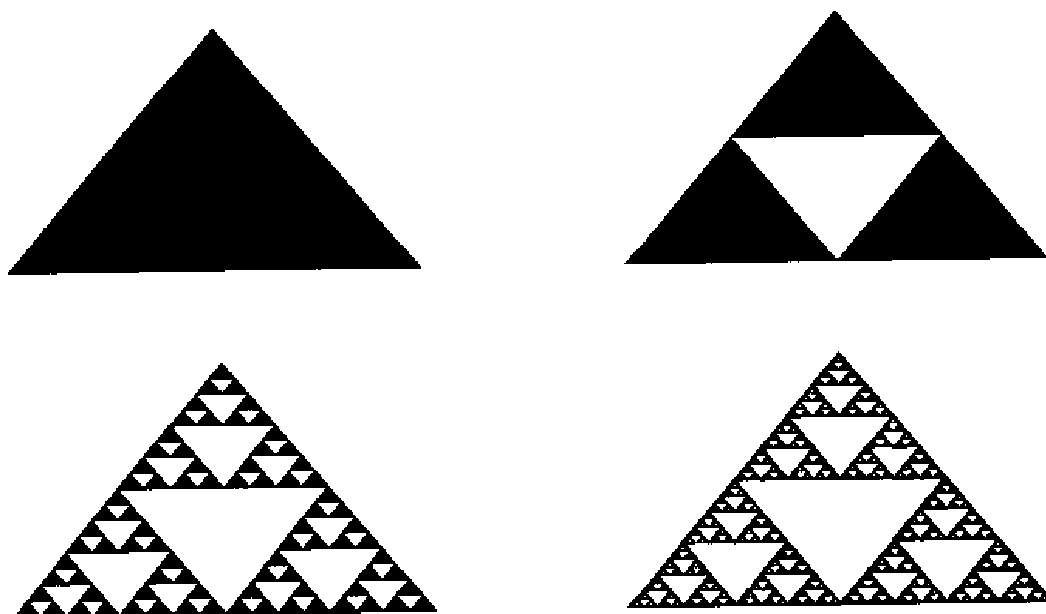


图 12—3 Sierpinski 三角形

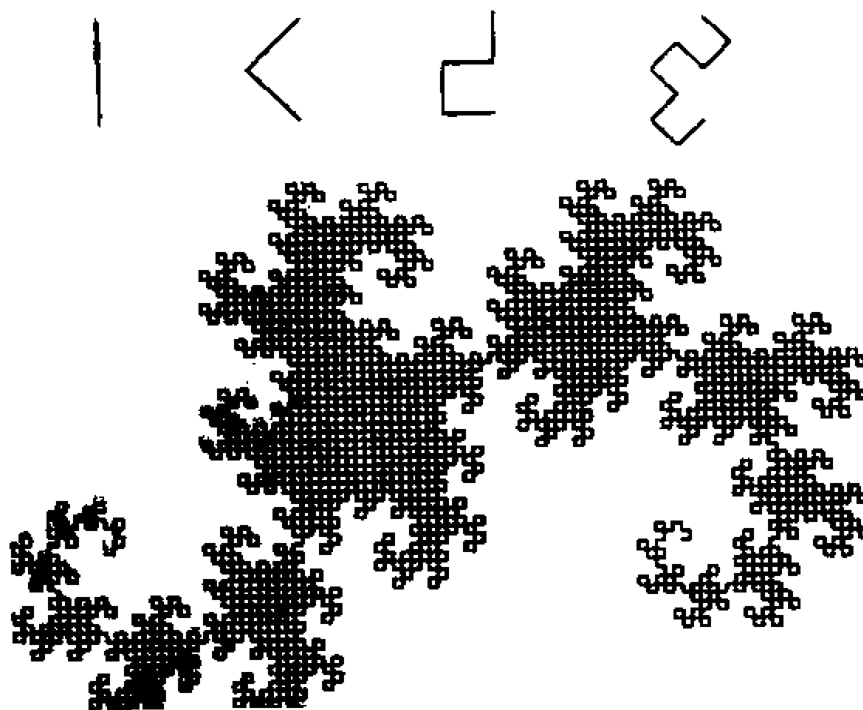


图 12—4 龙曲线

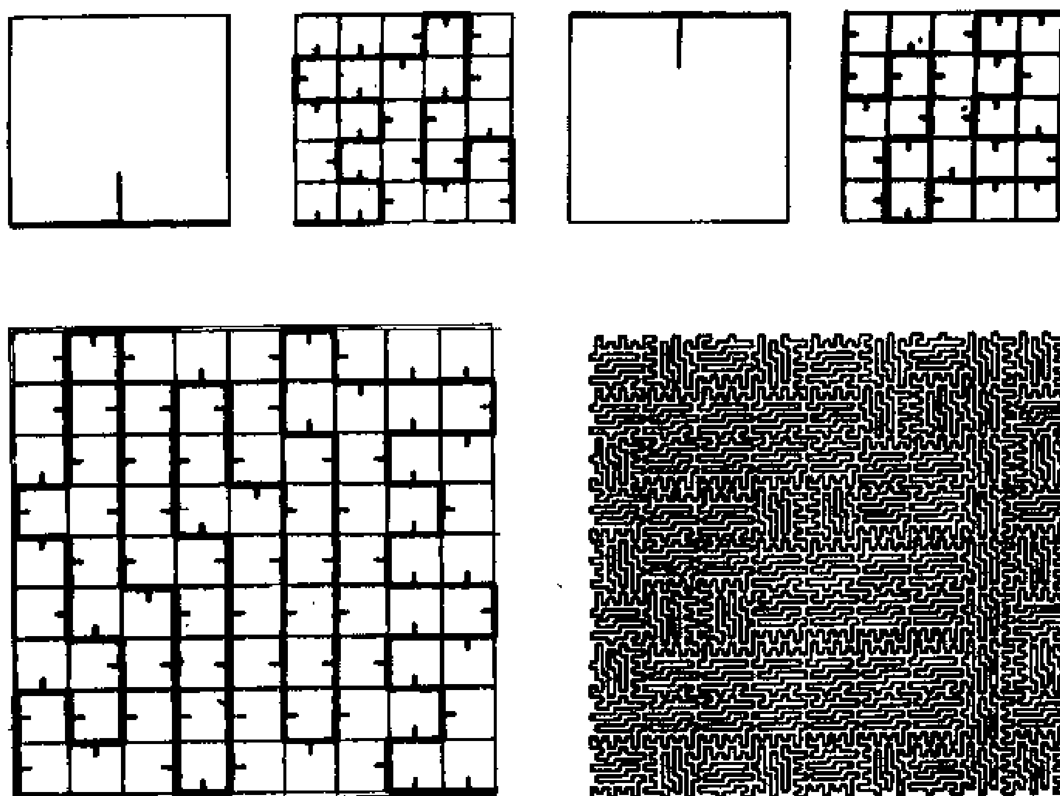


图 12—5 Hilbert 曲线

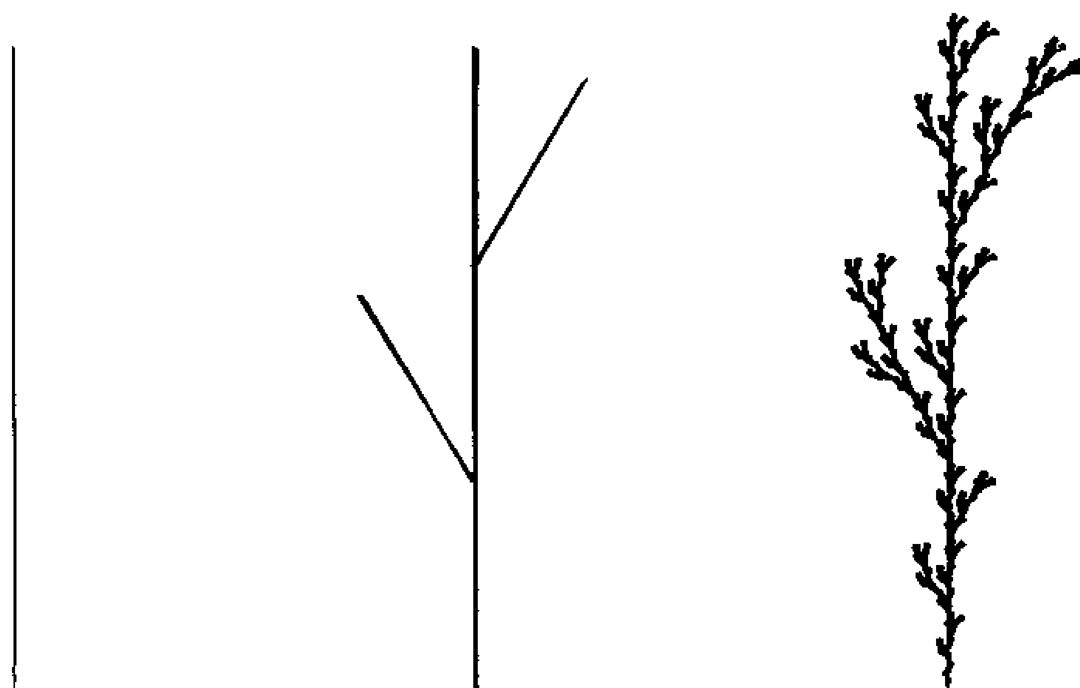


图 12 6 树木花草

其中，前两种分形图形的生成元比较简单．但后三种分形图形的生成元相对比较复杂．在龙曲线中，生成元将一直线段修改为由两互相垂直的线段构成的折线．但在决定向哪个方向折时存在两种选择（假设线段都有确定的定向，即我们要决定折线是在线段的左边还是在右边）．本生成元规定，折线方向对每条线段依次交替地改变（见图 12－4）．对 Hilbert 曲线，虽然它的生成元十分复杂，但其原理与龙曲线类似．在图 12－5 中，每个小线段的左侧或右侧都画了一根短线，它并不是分形图形的组成部分，它表示在下一步迭代时，生成元应位于短线指示的小正方形之内．树木花草的生成元有些特别，它具有所谓的分支结构．其中有一些参数可以改变，如每段树枝的长度以及树枝之间的夹角 δ．

练习 1

用计算机绘出 Koch 曲线，Sierpinski 三角形及一树木花草的图形．你能否自己构造一些生成元，并由此绘出相应的分形图形？任取分形图形的一个局部并将它放大，它同原来的分形图形有什么关系？

练习 2

从一个正三角形出发，用 Koch 曲线的生成元做迭代得到的极限图形称为 Koch 雪花曲线．

（1）试计算雪花曲线的边长及面积，它们是否有限？你如何解释所得出的结论？

（2）雪花曲线是否光滑（即每一点是否有切线存在）？

（3）其它的一些分形是否具有类似的性质？

练习 3

假设生成元由 n 条线段构成，每段长度为原线段长度的 $\frac{1}{c}$，定义该分形的维数为

$$d = \frac{\log(n)}{\log(c)}$$

（1）想一想，这样定义分形的维数有什么道理？分形维数的大小反映了分形的什么特性？

（2）利用上述公式，计算雪花曲线，Minkowski 香肠的维数．

（3）从直观上看，Hilbert 曲线将填满整个正方形，你是否相信这个事实？

你猜测 Hilbert 曲线的维数是少?

(4)Sierpinski 三角形的维数该如何定义?按你的定义,计算 Sierpinski 三角形的维数,并解释所得结果.

练习 4

定义 Weierstrass 函数如下:

$$W(x) = \sum_{k=1}^{\infty} \lambda^{(s-2)k} \sin(\lambda^k x), \quad \lambda > 1, \quad 1 < s < 2$$

对不同的 s 值,画出函数的图象.观察图象的不规则性与 s 的关系,由此猜测 Weierstrass 函数图象的维数与 s 的关系.

12.2 复变函数迭代

早在上个世纪就有一些数学家对复变函数的迭代进行研究.然而,直到本世纪 80 年代,B. Mandelbrot 才将复变函数的迭代与分形联系起来,并绘制出了第一张以他的名字命名的引人入胜的分形图形.复变函数的迭代由此再一次成为数学家的热点研究问题.

给定初始复数 Z_0,考虑如下的迭代

$$Z_{k+1} = Z_k^2 + \mu, \quad k = 0,1,2,\cdots \tag{1}$$

其中 Z_k, $k = 1,2,\cdots$, 为复数, μ 为(复)常数.

对于给定的初始点 Z_0,迭代序列 $\{Z_k\}_{k=0}^{\infty}$ 有可能有界,也可能发散到无穷.令 J_μ 是使得迭代序列 $\{Z_k\}_{k=0}^{\infty}$ 有界的所有初值 Z_0 构成的集合,即

$$J_\mu = \{Z_0 \mid \text{迭代序列}\{Z_k\}_{k=0}^{\infty}\text{有界}\} \tag{2}$$

我们称 J_μ 在复平面上构成的集合为 Julia 集.对不同的参数 μ,Julia 集的形状也会不同.特别地,$\mu = 0$ 对应的 Julia 集为单位圆盘.

如果固定初值 Z_0,则对不同的参数 μ,迭代序列 $\{Z_k\}_{k=0}^{\infty}$ 的有界性也不相同.令 M_{Z_0} 是使得迭代序列 $\{Z_k\}_{k=0}^{\infty}$ 有界的所有参数值 μ 构成的集合,即

$$M_{Z_0} = \{\mu \mid \text{迭代序列}\{Z_k\}_{k=0}^{\infty}\text{有界}\} \tag{3}$$

则称 M_{Z_0} 在复平面上构成的集合为 Mandelbrot 集.

为了便于在计算机上绘制出 Julia 集与 Mandelbrot 集,我们令 $Z_k = x_k + \mathrm{i}y_k$, $\mu = p + \mathrm{i}q$,则(1)可改写为

$$\begin{cases} x_{k+1} = x_k^2 - y_k^2 + p \\ y_{k+1} = 2x_k y_k + q \end{cases} \quad k = 0,1,2,\cdots \tag{4}$$

记 $r_k = x_k^2 + y_k^2$,则 Julia 集为使得序列 $\{r_k\}_{k=0}^{\infty}$ 有界的初始点 (x_0, y_0) 构成的集合,Mandelbrot 集为使得序列 $\{r_k\}_{k=0}^{\infty}$ 有界的参数 (p,q) 构成的集合.Julia 集与 Mandebrot 集会是什么样子?如果没有计算机的帮助,你是很难想象的.下面,我们给出这两种集合的计算机做图方法.

Julia 集绘制方法:

(1)设定初值 p,q ,一个最大的迭代次数 N ,图形的分辨率的大小 a,b 和使用的颜色数 K (如 $K = 16$)(或者给定灰度级 L).

(2)设定一个上界值 $M \geqslant \max(2, \sqrt{p^2 + q^2})$.

(3)将矩形区域 $R:= \{(x,y) \mid -M \leqslant x, y \leqslant M\}$ 分成 $a \times b$ 的网格,分别以每个网格点 (f_i, g_j) , $f_i = -M + \frac{2M}{a} \times i$, $g_j = -M + \frac{2M}{b} \times j$, $i = 0,1,\cdots,a$, $j = 0,1,\cdots,b$ 作为初值 (x_0, y_0) 利用(riter)做迭代(实际上,只需对满足 $x_0^2 + y_0^2 \leqslant M^2$ 的初始点迭代).如果对所有 $n \leqslant N$, $x_n^2 + y_n^2 \leqslant M^2$,则将图形的 (i,j) 象素点用黑色显示.否则,如果从迭代的某一步 n_0 开始有 $x_{n_0}^2 + y_{n_0}^2 > M^2$,则用第 $n_0 \bmod K$ 种颜色显示相应象素(或者用相应的灰度级显示).

Mandelbrot 集的绘制方法:

(1)设定一个最大的迭代次数 N ,图形的分辨率的大小 a,b 和使用的颜色数 K (如 $K = 16$)(或者给定灰度级 L).

(2)设定一个上界值 $M \geqslant 2$.

(3)将矩形区域 $R:= \{(p,q) \mid -M \leqslant p,q \leqslant M\}$ 分成 $a \times b$ 的网格,分别以每个网格点 (f_i, g_j) , $f_i = -M + \frac{2M}{a} \times i$, $g_j = -M + \frac{2M}{b} \times j$, $i = 0,1,\cdots,a$, $j = 0,1,\cdots,b$ 作为参数值 (p,q) 利用(riter)做迭代(实际上,只需对满足 $p^2 + q^2 \leqslant 4$ 的初给点迭代),每次迭代的初值均取为 $(x_0, y_0) = (0,0)$.如果对所有 $n \leqslant N$, $x_n^2 + y_n^2 \leqslant M^2$,则将图形的 (i,j) 点用黑色显示.否则,如果从迭代的某一步 n_0 开始有 $x_{n_0}^2 + y_{n_0}^2 > M^2$,则用第 $n_0 \bmod K$ 种颜色显示相应象素(或者用相应的灰度级显示).

练习 5

编写绘制 Julia 集的程序.对不同的参数 (p,q) : $(0,1)$, $(-1,0)$, $(0.11, 0.66)$, $(-0.102\,81, 0.957\,23)$, $(-1.25, -0.01)$ 观察 Julia 集的变化.取 Julia 集的不同局部放大,你能看到某种自相似现象吗?

练习 6

绘制 Mandelbrot 集. 然后,任意选取它的一个局部将其放大,然后再将放大图形的不同局部放大. 由此观察 Mandelbrot 集与 Julia 集有何关系. 进一步,取参数 μ 位于 Mandelbrot 集的不同部位(如内部、外部、边界、某个芭芽的内部等),观察相应的 Julia 集的形状的变化.

练习 7

Julia 集与 Mandelbrot 集可以推广到高阶情形. 一般地,考虑迭代

$$Z_{k+1} = Z_k^n + c, \qquad k = 0,1,\cdots,$$

使得迭代(5)有界的初值 Z_0 构成 n 阶 Julia 集. 对固定的参数 Z_0,使得迭代(5)有界的参数 c 构成 n 阶 Mandelbrot 集. 试画出三阶、四阶及五阶 Julia 集与 Mandelbrot 集,并通过放大不同的局部观察它们的自相似性.

10.3 IFS 迭代

让我们以下面的一个简单的绘图游戏来说明 IFS 迭代.

在平面上任取不共线的三点 A, B 和 C,并确定三个相应的概率 $p_A = 0.50$, $p_B = 0.47$, $p_C = 0.03$. 任取一个点 Z_0,按下述步骤在平面上画出新的点列 Z_n, $n=1,2,\cdots$,:Z_{n+1} 以概率 p_A 选定为 A 与 Z_n 的中点,以概率 p_B 选定为 B 与 Z_n 的中点,以概率 p_C 选定为 C 与 Z_n 的中点. 即

$$Z_{n+1} = \begin{cases} \dfrac{Z_n + A}{2} & \text{以概率} p_A \\ \dfrac{Z_n + B}{2} & \text{以概率} p_B \\ \dfrac{Z_n + C}{2} & \text{以概率} p_C \end{cases}$$

如果我们按上述步骤迭代下去,最终得到的图形会是什么样子呢?如果迭代仅仅只有一千次或一万次,你将会看到乱七八糟的一片. 因此,有人把这个游戏称为"混沌游戏". 但当迭代次数在一百万次以上时,图形将渐渐开始清晰. 图 12—7 给出了迭代一亿次的结果. 读者可以看出,它正是我们前面提到的 Sierpinski 三角形的逼近图形!

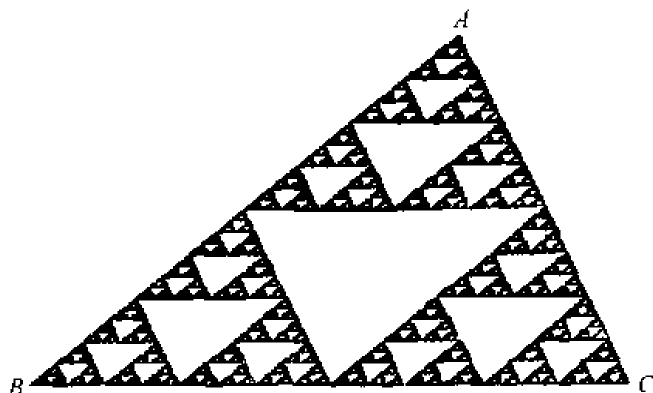


图 12－7

上述绘图游戏中的迭代即是 IFS 迭代. IFS 迭代的一般提法是：

给定一组(仿射)变化 w_i：

$$w_i(x,y)=(a_{11}^ix+a_{12}^iy+b_1^i,a_{21}^ix+a_{22}^iy+b_2^i),\quad i=1,2,\cdots,n$$

以及相应的一组概率 $p_1,\cdots,p_n$ ($p_1+\cdots+p_n=1$, $p_i>0$). 对于任意选取的初始值 $Z_0=(x_0,y_0)$ ，以概率 p_i 选取变换 w_i 做迭代

$$Z_{n+1}=(x_{n+1},y_{n+1})=w_i(x_n,y_n),\qquad n=0,1,\cdots$$

则点列 $Z_n,n=0,1,\cdots$ 收敛的极限图形称为一个 IFS 吸引子. 利用 IFS 迭代不仅可以生成许多有趣的分形图形，而且可以应用于分形图象的压缩. 下面，我们给出 IFS 迭代绘制分形的方法.

设计算机屏幕的可视窗口为

$$V=\{(x,y)\mid x_{\min}\leqslant x\leqslant x_{\max},\quad y_{\min}\leqslant y\leqslant y_{\max}\}$$

按分辨率大小的要求将 V 分成 $a\times b$ 的网格，网格点为 (x_i,y_j) ，这里

$$x_i=x_{\min}+(x_{\max}-x_{\min})i/a,\quad i=0,1,\cdots,a$$

$$y_j=y_{\min}+(y_{\max}-y_{\min})j/b,\quad j=0,1,\cdots,b$$

用 V_{ij} 表示矩形区域 $\{(x,y)\mid x_i\leqslant x\leqslant x_{i+1},\quad y_j\leqslant y\leqslant y_{j+1}\}$. 假设我们采取具有 L (如 $L=256$)级灰度的黑白图象绘制，总共的迭代次数为 N ，其中落于区域 V_{ij} 中的点的个数为 μ_{ij} . 再记

$$\mu=\max\mu_{ij},\quad i=0,1,\cdots,a-1,\quad j=0,1,\cdots,b-1$$

则象素 (i,j) 的灰度 $G(i,j)$ 与落于 V_{ij} 中的点数成正比：

$$G(i,j)=-\mu_{ij}\,\mu\times L$$

于是即给出了 IFS 迭代产生的分形的 L 级灰度图象.

练习 8

回到前面的"混沌游戏".如果把三个概率分别取为 $p_A = p_B = p_C = \frac{1}{3}$,则迭代一百万次的图形如何?再取一组其它的概率值绘制相应的图形,结果又如何?据此,你认为概率的选取对最终的极限图形(IFS 吸引子)有没有影响?概率的作用是什么?

练习 9

给定仿射变换

$$w_1(Z) = sZ + 1, \quad w_2(Z) = sZ - 1$$

以及相应的概率 $p_1 = p_2 = \frac{1}{2}$,其中 s,Z 均为复数.取 $s = 0.5 + 0.5\mathrm{i}$,绘制相应 IFS 迭代的吸引子的图形.取不同的 s,观察图形的变化.

练习 10

考虑两个更一般的变换

$$w_1(Z) = \sqrt{Z - C}, \qquad w_2(Z) = \sqrt{Z + C}$$

以及相应的概率 $p_1 = p_2 = \frac{1}{2}$,其中 C,Z 为复数.取不同的参数 C(如 $C = \mathrm{i}$,$C = -1$,$C = 0.11 + 0.66\mathrm{i}$ 等).观察图形的变化.你妮发现什么现象?

附录 1 Mathematica 程序

1. Koch 雪花曲线

```
redokoch[ptlist_List] :=
  Block[{tmp = {},i,pnum = Length[ptlist]},
    For[i = 1,i < pnum,i = i + 1,tmp = Join[tmp,{ptlist[[i]],
      ptlist[[i]] * 2/3 + ptlist[[i + 1]]/3,
    (ptlist[[i]] + ptlist[[i + 1]])/
    2 + {ptlist[[i]][[2]] - ptlist[[i + 1]][[2]],
    ptlist[[i + 1]][[1]] - ptlist[[i]][[1]]} * Sqrt[3]/6,
```

```
    ptlist[[i]]/3 + ptlist[[i + 1]] * 2/3,
    ptlist[[i + 1]]}]];tmp]

lnko01 = {{0,0},{1,0}};

Show[Graphics[Line[Nest[redokoch,lnko01,5]]],
            AspectRatio -> Sqrt[3]/6]]
```

2. Minkowski 香肠

```
redominkowski[ptlist_List] :=
  Block[{tmp = {},tmp1,i,pnum = Length[ptlist]},
    For[i = 1,i < pnum,i = i + 1,
      tmp1 = {ptlist[[i]][[2]] - ptlist[[i + 1]][[2]],
        ptlist[[i + 1]][[1]] - ptlist[[i]][[1]]}/4;
      tmp = Join[tmp,{ptlist[[i]],
    ptlist[[i]] * 3/4 + ptlist[[i + 1]]/4,
        ptlist[[i]] * 3/4 + ptlist[[i + 1]]/4 + tmp1,
        ptlist[[i]]/2 + ptlist[[i + 1]]/2 + tmp1,
    ptlist[[i]]/2 + ptlist[[i + 1]]/2,
    ptlist[[i]]/2 + ptlist[[i + 1]]/2 - tmp1,
    ptlist[[i]]/4 + ptlist[[i + 1]] * 3/4 - tmp1,
    ptlist[[i]]/4 + ptlist[[i + 1]] * 3/4,
  ptlist[[i + 1]]}]

    ];
  tmp
]

redomk1[ptlist_List] :=
Block[{tmp = {ptlist[[1]][[2]] - ptlist[[2]][[2]],
    ptlist[[2]][[1]] - ptlist[[1]][[1]]}/4},
    {ptlist[[1]],
    ptlist[[1]] * 3/4 + ptlist[[2]]/4,
```

```
        ptlist[[1]] * 3/4 + ptlist[[2]]/4 + tmp,
        ptlist[[1]]/2 + ptlist[[2]]/2 + tmp,
        ptlist[[1]]/2 + ptlist[[2]]/2,
        ptlist[[1]]/2 + ptlist[[2]]/2 - tmp,
        ptlist[[1]]/4 + ptlist[[2]] * 3/4 - tmp,
        ptlist[[1]]/4 + ptlist[[2]] * 3/4,
        ptlist[[2]]
    }]

    redomk2[ptlist_List] :=
      Block[{tmp = {},i,pnum = Length[ptlist]},
        For[i = 1,i < pnum,i = i + 1,
            tmp = Join[tmp,redomk1[{ptlist[[i]],
        ptlist[[i + 1]]}]]];
      tmp
    ]

    ln01 = {{0,0},{1,0}};

    Show[Graphics[Line[Nest[redominkowski,ln01,4]]],
              AspectRatio -> 1/GoldenRatio]]
```

3. Sierpinski 三角形

```
    redosierpinski[ptlist_List] :=
    Block[{tmp = {},i,pnum = Length[ptlist]/3},
        For[i = 0,i < pnum,i = i + 1,
        tmp = Join[tmp,{ptlist[[3i + 1]],
            (ptlist[[3i + 1]] + ptlist[[3i + 2]])/2,
      (ptlist[[3i + 1]] + ptlist[[3i + 3]])/2,
            (ptlist[[3i + 1]] + ptlist[[3i + 2]])/2,
            ptlist[[3i + 2]],
            (ptlist[[3i + 2]] + ptlist[[3i + 3]])/2,
            (ptlist[[3i + 1]] + ptlist[[3i + 3]])/2,
```

```
            (ptlist[[3i + 2]] + ptlist[[3i + 3]])/2,
    ptlist[[3i + 3]]}]];

tmp]

showsierpinski[ptlist_List] :=
Block[{tmp = {},i,pnum = Length[ptlist]/3},
    For[i = 0,i < pnum,i = i + 1,
    AppendTo[tmp,
            Polygon[{ptlist[[3 * i + 1]],
ptlist[[3 * i + 2]],ptlist[[3 * i + 3]]}]]];
      Show[Graphics[tmp],AspectRatio -> 1/GoldenRatio]]

po1 = {{- 1,0},{1,0},{0,Sqrt[3]}};

showsierpinski[Nest[redosierpinski,po1,4]]
```

4. 树木花草

```
redotree[ptlist_List] :=
Block[{tmp = {},i,ptnum = Length[ptlist]/2,
      midpt1,midpt2,leftpt,rightpt},
    For[i = 0,i < ptnum,i = i + 1,
      midpt1 = (ptlist[[2i + 1]] * 2 + ptlist[[2i + 2]])/3;
      midpt2 = (ptlist[[2i + 1]] + ptlist[[2i + 2]] * 2)/3;
      leftpt = midpt1 + {{Cos[theta], - Sin[theta]},
      {Sin[theta],Cos[theta]}}.{
        ptlist[[2i + 2]][[1]] - ptlist[[2i + 1]][[1]],
        ptlist[[2i + 2]][[2]] - ptlist[[2i + 1]][[2]]}/3;
      rightpt = midpt2 + {{Cos[theta],Sin[theta]},
      {- Sin[theta],Cos[theta]}}.{
        ptlist[[2i + 2]][[1]] - ptlist[[2i + 1]][[1]],
        ptlist[[2i + 2]][[2]] - ptlist[[2i + 1]][[2]]}/3;
    tmp = Join[tmp,{ptlist[[2i + 1]],midpt1,midpt1,leftpt,
```

```
        midpt1,midpt2,midpt2,rightpt,
      midpt2,ptlist[[2i + 2]]}]];
    tmp]

    showtree[ptlist_List] := 
    Block[{tmp = {},i,ptnum = Length[ptlist]/2},
        For[i = 0,i < ptnum,i = i + 1,
            AppendTo[tmp,
        Line[{ptlist[[2i + 1]],ptlist[[2i + 2]]}]]];
      Show[Graphics[tmp],AspectRatio -> 3/2/Sin[theta]]
      ]

    theta = 30Degree;

    showtree[Nest[redotree,{{0,0},{0,1}},4]]
```

5. Weierstrass 函数

```
    lambda = 2;nmax = 20;s = 1.5;
    Plot[Sum[lambda ^ ((s - 2)k)Sin[lambda ^ kx],
            {k,1,nmax}],{x, - 1,1}]
```

6. Mandelbrot 集以及它的局部放大

```
    iter[x_,y_,lim_] := Block[{c,z,ct},
    c = x + I * y;z = c;ct = 0;
    While[(Abs[z] < 2.0)&&(ct < lim),
           ++ ct;z = z * z + c;];
    Return[ct];]

    Mandelbrot1 = 
    DensityPlot[iter[x,y,50],{x, - 2.0,1.0},{y, - 1.5,1.5},
              PlotPoints -> 120,Mesh -> False]
```

```
Mandelbrot2 = Show[Mandelbrot1,
            Graphics[Line[{{- 0.9, - 0.25},
  {- 0.7, - 0.25},{- 0.7, - 0.05},
  {- 0.9, - 0.05},{- 0.9, - 0.25}}]]]

Mandelbrot3 = DensityPlot[iter[x,y,50],
            {x, - 0.9, - 0.7},{y, - 0.25, - 0.05},
      PlotPoints -> 120,Mesh -> False]
```

7. Julia 集以及它的局部放大

```
julia[x_,y_,lim_,cx_,cy_] :=
  Block[{z,ct = 0},z = x + I * y;
    While[(Abs[z] < 2.0)&&(ct < lim), ++ ct;
          z = z * z + (cx + I * cy);];Return[ct];]

julia1 = DensityPlot[julia[x,y,50,0.27334,0.00742],
          {x, - 1.5,1.5},{y, - 1.5,1.5},
          PlotPoints -> 120,Mesh -> False]

julia2 = Show[julia1,Graphics[Line[{{- 0.7, - 0.1},
            {- 0.3, - 0.1},{- 0.3,0.3},
   {- 0.7,0.3},{- 0.7, - 0.1}}]]]

julia3 = DensityPlot[julia[x,y,50,0.27334,0.00742],
            {x, - 0.7, - 0.4},{y, - 0.1,0.3},
            PlotPoints -> 120,Mesh -> False]
```

8. IFS 迭代生成 Sierpinski 三角形

```
p1 = 0.3;aaa = 1/2 + I1//N;      f1[z_] := (z + aaa)/2;
p2 = 0.3;bbb = 0//N;             f2[z_] := (z + bbb)/2;
```

```
p3 = 0.4;ccc = 1//N;              f3[z_] := (z + ccc)/2;

f[z_] := Block[{tmp},tmp = Random[];
  Which[tmp < p1,f1[z],tmp < p1 + p2,f2[z],True,f3[z]]];

Array[mu,{150,150}];
showIFS[z0_,shrange_List,divi_List,nmax_] :=
  Block[{i,j,z = z0,a = divi[[1]],b = divi[[2]],
      temp1,temp2,mumax = 0},
    For[i=a,i>=1,i --,For[j=b,j>=1,j --,mu[i,j]=0]];
    For[i = nmax,i >= 1,i --,
        temp1 = Floor[divi[[1]] * (Re[z] - shrange[[1]][[1]])
        /(shrange[[2]][[1]] - shrange[[1]][[1]])] + 1;
    temp2 = Floor[divi[[2]] * (Im[z] - shrange[[1]][[2]])
    /(shrange[[2]][[2]] - shrange[[1]][[2]])] + 1;
    mu[temp1,temp2] ++;
    z = f[z];
  ];
  For[i = a,i >= 1,i --,
    For[j=b,j>=1,j --,mumax=Max[mumax,mu[i,j]]]];
  mu1=Table[GrayLevel[1-N[mu[j,i]]/mumax],{i,a},{j,b}];

  Show[Graphics[RasterArray[mu1]]]
    ]

showIFS[0+I0,{{- 0.1,- 0.1},{1.1,1.1}},{150,150},100000]
```

附录 2 Mandelbrot 集与 Julia 集的 C 程序

```
#include "graphics.h"
#include "stdio.h"
```

```
#include "conio.h"
#include "stdlib.h"
#include "math.h"
#include "time.h"

#define SCREEN_WIDTH 800
#defineSCREEN_HEIGHT 600

/*****************************
Mandelbrot set
*****************************/
#define MAXCOLOR 16
#define MOD 100
#define SUM 30000
void DrawMandelbrot(int x_top,int y_top,
                    int x_bottom,int y_bottom)
{
int np,nq,k;
int ViewportX=x_bottom-x_top,
    ViewportY=y_bottom-y_top;
double x[MAXCOLOR+1],y[MAXCOLOR+1];
double r,p,q,pmin=-2.25,pmax=.75,qmin=-1.5,qmax=1.5;
double deltap=(pmax-pmin)/(double)(ViewportX-1),
        deltaq=(qmax-qmin)/(double)(ViewportY-1);
for(np=0;np<ViewportX;np++)
        for(nq=0;nq<ViewportY;nq++)
{
k=0;
p=pmin+np*deltap;
q=qmin+nq*deltaq;
x[0]=y[0]=0;
do
{
x[k+1]=x[k]*x[k]-y[k]*y[k]+p;
y[k+1]=2*x[k]*y[k]+q;
```

```
k++;
r=x[k]*x[k]+y[k]*y[k];
}while(k<MAXCOLOR&&r<=MOD);
if(r>MOD) putpixel(np+x_top,nq+y_top,k);
if(k==MAXCOLOR)putpixel(np+x_top,nq+y_top,0);
    }
}


/*****************************
Julia set
*****************************/
void DrawJuliaSet(int x_top,int y_top,int x_bottom,
                int y_bottom,double p,double q)
{
int nx,ny,k=0;
int ViewportX=x_bottom-x_top,
    ViewportY=y_bottom-y_top;
double r;
double x[MAXCOLOR+1],y[MAXCOLOR+1];
double xmin=-1.5,xmax=1.5,ymin=-1.5,ymax=1.5;
double deltax=(double)(xmax-xmin)/(double)(ViewportX-1),
      deltay=(double)(ymax-ymin)/(double)(ViewportY-1);
for(nx=0;nx<=ViewportX;nx++)
      for(ny=0;ny<=ViewportY;ny++)
      {
      k=0;
        x[0]=xmin+nx*deltax;
        y[0]=ymin+ny*deltay;
        do{

      x[k+1]=x[k]*x[k]-y[k]*y[k]+p;
      y[k+1]=2*x[k]*y[k]+q;
        k++;
        r=x[k]*x[k]+y[k]*y[k];
```

```
}while(k<MAXCOLOR && r<=MOD);
  if(r>MOD) putpixel(nx+x_top,y_bottom-ny,k);
  if(k==MAXCOLOR) putpixel(nx+x_top,y_bottom-ny,0);
    }
}


/* * * * * * * * * * * * * * * * * * * * * * * * * * *
Main Project
* * * * * * * * * * * * * * * * * * * * * * * * * * */
void main()
{
int gdriver=DETECT,gmode,errorcode;
/* initialize graphics mode */
initgraph(&gdriver,&gmode,"");
/* read result of initialization */
errorcode = graphresult();
if (errorcode != grOk) /* an error occurred */
{
  printf("Graphics error: %s\n",grapherrormsg(errorcode));
  printf("Press any key to halt:");
  getch();
  exit(1); /* return with error code */
}
  DrawMandelbrot(0,0,400,400);
  getch();cleardevice();
  DrawJuliaSet(0,0,200,200,-0.12,0.74);
  DrawJuliaSet(200,0,400,200,-1,-0.25);
  DrawJuliaSet(0,200,200,400,0.11031,-0.67037);
  DrawJuliaSet(200,200,400,400,0.35,0.05);
  getch();

}
```

参考读物

1. 齐东旭. 分形及其计算机生成. 北京:科学出版社,1994

2. 王东生,曹磊. 混沌、分形及其应用. 北京:中国科学技术大学出版社,1995

实验十三　迭代(三)——混沌

在实验九、实验十二中我们看到，利用一些简单的迭代格式可以求解方程(组)，同时还可以产生非常复杂而漂亮的分形图形. 在本实验中我们将看到，迭代还可以产生类似于随机行为的一种非常古怪的现象，我们把这种现象称之为混沌.

实际上，混沌与分形是密不可分的. 混沌中包含着分形，分形中包含着混沌. 它们是一对孪生兄弟. 本实验我们将从从一个简单的二次函数的迭代出发，使读者认识混沌现象及其所蕴涵的规律性.

13.1　周期点与周期轨道

在系统探讨混沌现象之前，我们约定一些基本的术语.

设 $f(x)$ 是一个定义在实数域上的实值函数，如果存在 u 使得 $f(u)=u$，则称 u 为 $f(x)$ 的**不动点**. 我们用 $u\to u$ 表示这件事. 如果所有附近的点在迭代过程中都趋向于某个不动点，则该不动点称为**吸引点**，有时也称该不动点是**稳定的**. 如果所有附近的点在迭代过程中都远离它而去，则该不动点称为**排斥点**，有时也称该不动点是**不稳定的**. 例如，0 与 1 是 $f(x)=x^2$ 的不动点. 0 是吸引点，因为从 $-1<x_0<1$ 的初值出发，迭代序列都收敛到 0. 而 1 是一个排斥点，因为从 $0<x_0<1$ 的初值出发，迭代序列收敛到 0，而从 $x_0>1$ 的初值出发，迭代序列发散到无穷.

如果 $f(u_1)=u_2, f(u_2)=u_3, \cdots, f(u_k)=u_1$，且 $u_j\neq u_1$，$j=2,\cdots,k$，则 $u_1,u_2,\cdots,u_k$ 形成一个 k 循环. 我们用 $u_1\to u_2\to\cdots\to u_k\to u_1$ 记这个事实. u_1 称为一个 k **周期点**，$u_1,u_2,\cdots,u_k$ 称为一个**周期轨道**. 显然，不动点就是周期为 1 的周期点. 类似于不动点，如果所有附近的点在迭代过程中都趋向于某个周期点，则该周期点称为**吸引点**. 如果所有附近的点在迭代过程中都远离它而去，则该周期点称为**排斥点**. 如果点 u 最终落于某个循环之中，则称它是一个**预周期点**. 例如，1 是 $f(x)=x^2-1$ 的预周期点.

13.2 二次函数的迭代

我们将从形如 $f(x)=ax(1-x)$ 的二次函数开始做迭代

$$x_{k+1}=f(x_k)=ax_k(1-x_k),\quad k=0,1,\cdots \tag{1}$$

这里，$a\in[0,4]$ 是一个参数.

为了从几何直观上观察迭代是否收敛，我们先按如下方法做出一个迭代的可视图形.

在平面直坐标系中，画出 $y=f(x)$ 及 $y=x$ 的函数图象. 在 x 轴上取一个初值 x_0，从 x_0 出发做平行于 y 轴的直线，它与 $y=f(x)$ 的图象相交于一点. 接着从该点出发做平行于 x 轴的直线与 $y=x$ 的图象相交于一点. 然后再从交点出发做平行于 y 轴的直线与 $y=f(x)$ 的图象相交于一点. 这个过程一直进行下去，你将得到一个类似阶梯或蜘蛛网的图形. 它正是函数 $y=f(x)$ 的迭代过程的几何形式.

练习 1

编程实现函数迭代的几何做图过程. 对几组不同的参数值 a（如 $a=0.5$，1，4）以及初值 x_0，观察迭代序列是否收敛，并比较结果.

练习 2

取参数 $a=0.8$，并使用不同的初值 x_0. 你能找出一个吸引的不动点吗？一个排斥的不动点吗？哪些初值收敛到吸引的不动点？哪些初值使序列发散到无穷？取不同的初值 $a=1$，$a=1.6$，$a=2$，$a=2.5$ 回答同样的问题.

练习 3

找出一个 a 值，它对应的迭代具有 2 周期点. 这种性质依赖于初值吗？你能找到多个 a 值具有这种性质吗？

练习 4

你能对任意的 k 找到一个 a 值，使得它对应的迭代具有 k 周期点吗？哪些 k 值能给出 k 周期点？在每种情况下，结果是否依赖于初值的选取？（请对

$3.4 \leqslant a \leqslant 3.6$ 和 $3.6 \leqslant a \leqslant 4$ 的值进行验证).

练习 5

如果某个 a 值能给出周期点,它是否一定是吸引的周期点?你能否找到排斥的周期点?

练习 6

根据前面的一系列练习,试着从理论上分析:$f(x)$ 的不动点是什么? 对哪些 a 值迭代收敛到每个不动点? 哪些初值收敛到不动点? 哪些初值导致发散? 对周期点做类似的分析.

13.3 Feigenbaum 图

本节我们对不同的 a 系统地观察迭代(1)的行为.

将区间 $(0,4]$ 以某个步长 Δa (如 $\Delta a = 0.04$)离散化.对每个离散的 a 值做迭代(1),忽略前 50 个迭代值 $x_1, x_2, \cdots, x_{50}$,而把点 $(a, x_{51}), (a, x_{52}), \cdots, (a, x_{100})$ 显示在坐标平面上.最后形成的图形称为 Feigenbaum 图.

练习 7

观察 Feigenbaum 图.在它的左部有一条曲线,这代表迭代具有唯一的吸引不动点.

(1)从某一点 a_1 开始,该条曲线分成两条曲线,这说明了迭代的什么性质? 迭代的点列是如何运动的?

(2)再在下一个分支点 a_2 ,曲线分成几个分支? 这说明迭代的什么性质? 相应的迭代点列是如何运动的?

(3)上述分支过程是否一直进行下去? 是否存在一个极限的分支点 a_∞ ? 在极限分支点之后,Feigenbaum 图是否显得很混乱?

(4)估计出 a_n, $n = 1,2,3,4,5$ 等的值,再计算比值 $\delta_n = (a_n - a_{n-1})/(a_{n+1} - a_n)$, $n = 2,3,4$ 等.它们是否比较接近?由此猜测数列 δ_n 是否会收敛?

练习 8

在 Feigenbaum 图的右部，你应当能看到一个由三条曲线穿过的空白带，它是一个"周期为 3 的窗口". 你能找到其它窗口吗？它们的周期是什么？窗口里有什么图案？这些窗口跟练习 4 中的 k 周期轨道有什么关系？

13.4　混沌的特性

从极限分支点之后，Feigenbaum 图显得很杂乱，似乎没有任何规律. 实际上，对任何初始值做迭代都会得到同样的结果. 这就是所谓的混沌现象. 迄今为止，混沌并没有确切的数学定义. 但它具有一些基本的特性，如对初值的敏感性以及某种无序性，由此产生类似于随机的现象. 下面，我们以 $a = 4$ 为例来说明迭代(1)具有这些特性.

1. 对初值的敏感性

所谓一个迭代对初值是敏感的意思是，无论两个初值如何接近，在迭代过程中它们将渐渐分开. 这是任何一个混沌系统都具有的特性之一. 这种特性使得混沌系统会产生似乎是随机的、没有规律的现象. 下面，我们考察迭代

$$x_{n+1} = 4x_n(1 - x_n), \quad n = 0,1,\cdots \tag{2}$$

是否具有这种性质.

练习 9

任取两个初值使得它们之间的差的绝对值不超过 0.1，在迭代过程中它们逐渐分开吗？如果两个初值之间的差的绝对值不超过 0.01，0.001，结果会如何？由此得出，函数 $f(x) = 4x(1 - x)$ 的迭代对初值是否敏感？

2. 混沌不是随机的

从上面的实验可以看到，一个简单的、确定的二次迭代可以产生非常复杂的、看似随机的行为. 但是，混沌不等于随机. 实际上，在混沌区域之内，蕴涵着许多有序的规律. 这正验证了哲学上的名言：有序中包含了无序，无序中包含着有序.

练习 10

从不同的初值 $x_0 \in (0,1)$ 出发做迭代(2). 统计迭代点列中分别落于区间 $\left(0,\frac{1}{2}\right)$ 以及区间 $\left(\frac{1}{2},1\right)$ 中的点的个数. 你得到的结果是随机的吗? 进一步, 将 $(0,1)$ 区间分成任意等份, 统计迭代点列中分别落于不同的等份区间的点的个数. 你的结论又如何? 在区间 $[0,1]$ 的各种划分中, 是否有一些区间即使在大量的迭代之后也不包含任何迭代点?

另一个说明混沌不是随机的事实是, 混沌区域有许多有序的窗口. 将这些窗口放大可以看到令人振奋的自相似现象, 同时还有许多周期轨道.

练习 10 确定迭代(2)的一个 2 周期点. 该周期点是吸引点还是排斥点? 找出该周期点的一个有理逼近(误差尽量小)做初值, 利用(2)做迭代. 迭代结果是否象一个周期为 2 的循环轨道?

13.5 其它函数的迭代

和函数 $f(x) = 4x(1-x)$ 一样有着混沌行为的函数还很多. 其中较简单的有"帐篷函数"和"锯齿函数". "帐篷函数" $T(x)$ 定义为

$$T(x) = \begin{cases} 2x & \text{如果 } 0 \leqslant x \leqslant \frac{1}{2} \\ 2(1-x) & \text{如果} \frac{1}{2} < x \leqslant 1 \end{cases}$$

"锯齿函数" $S(x)$ 定义为

$$S(x) = \begin{cases} 2x & \text{如果 } 0 \leqslant x \leqslant \frac{1}{2} \\ 2x-1 & \text{如果} \frac{1}{2} < x \leqslant 1 \end{cases}$$

容易验证, 帐篷函数和锯齿函数有下列关系:

$$T(T(x)) = T(S(x)), \qquad 0 \leqslant x \leqslant 1 \tag{3}$$

练习 11

锯齿函数对初始条件是否敏感? 找出锯齿函数的的所有周期点(提示: 把数值表示成二进制形式).

练习 12

帐篷函数对初始条件是否敏感？找出帐篷函数的的所有周期点(提示：利用关系式(3).)

寻找帐篷函数和锯齿函数的周期点相对比较容易. 但要求得 $f(x)=4x(1-x)$ 得周期点就要困难许多. 如果有一种方法能够在较复杂的函数与较简单的函数之间进行变换，则我们可以借此找出那些难以捉摸的周期点. 实际上，如果令

$$h(x)=\sin^2(\pi x/2)$$

则

$$f^k(h(x))=h(T^k(x))$$

对任意正整数 k 成立.

练习 13

你能否利用上述关系式求出 $f(x)$ 的 4 周期点?该周期点是吸引的还是排斥的?

除了函数 $f(x)=4x(1-x)$ 外，还可以观察其它函数的迭代. 下面是其中的某几个函数. 有兴趣的读者可以对它们的迭代行为进行类似的探讨，并与 $f(x)$ 的迭代行为做比较.

1. $f(x)=a-(x-\sqrt{a})^2$
2. $f(x)=x^2-a$
3. $f(x)=a\sin(x)$
4. $f(x)=x^4-a$

13.6 听一听混沌

在本实验的最后，让我们来听一听混沌的一个吸引子，一个 2 循环，一个 4 循环的声音是什么样子的.

练习 14

选取 x_0 的一个初值(如 $x_0=0.1$)，利用(2)做迭代得到序列 $x_n, n=0,1,$

…，.根据 x_n 的大小确定相应的音调的高低.试编程按顺序演奏迭代序列.

附录　Mathematica 程序

本实验中的主要程序.

1. 迭代的几何图形.

```
IterGeo[u_,x0_]:=
  Module[
  {p1,p2,i,pointlist = {},var = x0,fvar = u * x0 * (1 - x0)},
  p1 = Plot[{u * x * (1 - x),x},{x,0,1},
    DisplayFunction -> Identity];
  AppendTo[pointlist,{x0,0}];
  For[i = 1,i < 20,i ++,AppendTo[pointlist,{var,fvar}];
      AppendTo[pointlist,{fvar,fvar}];
      var = fvar;fvar = u * var * (1 - var)];
  p2 = ListPlot[pointlist,PlotJoined -> True,
      DisplayFunction -> Identity];
  Show[{p1,p2},DisplayFunction -> $DisplayFunction]
  ]
```

2. Feigenbaum 图(将区间 [0,4]n 等分,**x0** 为迭代初值).

```
Feigenbaum[n_Integer,x0_]:=
Module[
{plist = {},a,i,temp,pilist = {}},
For[a = 1,a <= n,a ++,temp = x0;plist = {};
  For[i = 1,i <= 50,i ++,temp = 4 * a * temp * (1 - temp)/n];
  For[i=51,i <= 100,i ++,temp=4 * a * temp * (1-temp)/n;
        AppendTo[plist,{4a/n,temp}]];
  AppendTo[pilist,ListPlot[plist,
```

```
PlotStyle -> {PointSize[0.005]},
     DisplayFunction -> Identity]]
     ];
     Show[pilist,DisplayFunction -> $DisplayFunction]
       ]
```

3. 迭代(2)对不同初值的迭代结果的差.

```
Sensitivity[n_Integer,x01_,x02_] :=
  Module[
  {pilist = {},i,temp1 = x01,temp2 = x02},
  For[i = 1,i<= n,i ++,temp1 = 4 * temp1 * (1 - temp1);
        temp2 = 4 * temp2 * (1 - temp2);
  AppendTo[pilist,{i,temp2 - temp1}];
    ];
ListPlot[pilist,PlotJoined -> True]
]
```

4. 对初值 x_0 做 n 次迭代,迭代点在 m 个等分区间中的分布情况.

```
Distrib[n_Integer,m_Integer,x0_] :=
  Module[
  {i,temp = x0,count = Table[0,{i,m}]},
  For[i = 1,i<= n,i ++,temp = 4 * temp(1 - temp);
  If[temp==1,count[[m]]++,count[[Floor[temp * m]+1]]++]
  ];
  BarChart[count]
 ]
```

在应用此程序之间需要调用 Graphics 程序包,即要读人文件Graphics.m

5. 演奏混沌(对初值 x_0 做 n 次迭代,然后演奏所得的迭代数列).

```
PlayChaos[n_Integer,x0_]:=
  Module[
  {t = {},i,temp = x0},
  For[i = 1,i<= n,i ++,temp = 4 * temp * (1 - temp);
      AppendTo[t,Floor[temp * 100]]];
  ListPlay[t,PlayRange  > {0,100},SampleRate  > 5]
]
```

参考读物

1. Mount Holyoke College. 数学实验室. 白峰杉,蔡大用译. 北京:高等教育出版社;施普林格出版社,1998

2. 王东生,曹磊. 混沌、分形及其应用. 北京:中国科学技术大学出版社,1995

实验十四　密　　码

保密通讯在军事、政治、经济斗争和竞争中的重要性是不言而喻的.

在斗争或竞争中，一方要将信息传递给己方的接收者，同时又要防止其他人(特别是敌方)知道信息的内容. 他采用的一种方式是：将原来的信息(称为明文)经过加密，变成密文之后发送出去，使敌方即使得到密文也读不懂，而合法的接收者收到密文之后却可以按照预先约定好的方法加以解密，再翻译成明文. 而敌方却要千方百计从密文破译出明文来. 一方如何编制密码使之不易被破译，另一方则要找到其弱点加以破译，这就构成了密码学的主要内容.

14.1　单表密码

一提起加密，你也许就会想：一个人关在屋里编造出一个稀奇古怪的加密方法，不告诉别人，别人就不可能猜出来了. 下面的加密方案就很符合你这种想法：

单表密码：假定明文用英文撰写. 将明文中的每个字母用另一个预先指定的字母代替，就得到密文. 每一个字母用哪一个字母代替，你可以随心所欲，但必须预先规定好，并且让你的信息的合法接收者知道，他才能正确地解密. 而且，明文中不同的字母在密文中也必须不同，否则也无法解密. 也就是说，你的加密方案是将 26 个英文字母作一个置换 σ. 明文中的字母经过这么一替换，就变成一份面目全非的密文. 你的替换方案，既可以采用某种算法，也可以没有任何规律，只是随意制作一张对照表，按照这张表进行加密和解密. (需要指出：制作密码表并由加密者及接收者保存密码表，就产生了密码表被偷窃的危险.)26个字母的所有置换的个数多达 $26! \approx 10^{26}$，你从中任选一个，别人是不可能对如此多的置换一个一个审查来破译你的密码的. 这个密码方案岂不是很难攻破了吗？

为了看出单表密码方案的破绽，请先做以下练习.

练习 1

任意选一篇普通的英文文章.

(1)统计每个英文字母在文章中出现的次数.计算出每个字母出现的次数在总的字母个数中所占的百分比.再换几篇文章做同样的统计,比较统计的结果.

(2)统计每一种双字母组合出现的次数在总次数中所占的百分比,从中选出出现频率最高的双字母组合.

通过上面的练习,你会发现:各个英文字母在文章中出现的频率是不均衡的,甚至还有相当大的悬殊.同一个字母在不同文章中出现的频率当然不会完全相同,但也呈现出相当的稳定性,起伏不大.根据所统计的数据,就得出了破译单表密码的如下方法:对经过单表密码加密的密文中每个字母出现的频率以及双字母组合出现的频率进行统计,与英文字母在文章中出现频率的统计数据进行比较,就可能猜出密文字母代表哪个明文字母.比如:大量的统计结果表明,字母e在英文文章中出现的频率最高,约为12.7%.如果发现密文中某个字母出现的频率最高,接近12.7%,就很有理由认为这个字母代表明文字母e.对密文中其它字母也是这样,查看它在密文中的出现频率与哪个明文字母出现频率相近,这个密文字母就很有可能代表那个明文字母.这说明,由于英文字母在文章中出现频率不均衡,而单表密码将这种不均衡性保存了下来,就为破译这种密码提供了强有力的信息.当然,实际的破译过程没有这么简单,有可能由于密文不够长,统计规律不明显.有些字母的出现频率相差不大,难以区分.但只要有一部分出现频率与众不同的字母先被破译出来了,密文就已露出了破绽.再结合对于语言的知识以及对于密文的可能内容的猜测等等信息,经过不懈的努力,就有可能破译出密文的内容.这说明,单表密码有严重破绽,不是好的密码方案.

14.2 多表密码

鉴于单表密码的主要缺陷在于字母出现频率的不均衡性,密码编制者就想办法克服这一缺陷.办法之一是用多表密码代替单表密码.比如,他准备几张不同的密码表轮流使用.这样,明文中同一个字母被不同的密码表加密,被置换成不同的字母.而不同的字母通过加密之后反而可能变成同样的字母.字母出现频率的不均衡性被搅乱了,就可能比较均衡.比如,明文字母 e 在密文中被变成了几个不同的字母,它的高的出现频率也就被分散了,被隐藏起来了.这样,破译者就失去了可乘之机.鉴于制作并保存密码表就存在密码表被偷窃的危险,下面建议一种不用密码表的的多表密码方案.它是法国密码学家

维吉尼亚设计的，称为维吉尼亚密码.它采用加法密码作为单表密码，用不同的加法密码来构造多表密码.

加法密码：将 26 个英文字母从 a 到 z 按顺序排列，并且认为 z 后面是 a.秘密取定 0 到 25 的一个整数 α.将每个明文字母在英文字母表上向后移动 α 个位置，得到的字母作为密文字母.这样得到的密码叫做加法密码.每个加法密码由一个整数 α 决定，猜出了 α，就找到了攻破这个密码的钥匙.α 是加密和破译的关键，称为密钥.

比如，取 3 作为密钥.则明文的 a，b，c，…，w，x，y，z 依次被 d，e，f，…，z，a，b，c 就得到密文.（这是古罗马的将军凯撒曾经用过的密码.）

如果取 0 作密钥，就等于没有加密，这当然是不好的.即使不用 0 作密钥，加法密码本身也是不好的：密钥 α 只有 25 种可能性，一个一个穷举就可以攻破.但假如在加密过程中更换密钥，就不容易被破译了.但是，如果毫无规律地随意更换，则自己人也不能译出来，达不到传送信息的目的.最好是按照某种规律更换密钥，但这种规律不容易被人猜出来.维吉尼亚密码就是这样做的.

维吉尼亚密码：自己想好一个英文单词作为密钥.比如，用 codebreaking 作密钥.单词中的每个字母代表一个单表密码.比如，第一个字母 c，表示明文的 a 在密文中要变成 c，向后移动了两位，这就代表以 2 为密钥的加法密码.明文中的第一个字母要用这个加法密码加密，用它向后移动 2 位后的字母代替.第二个字母是 o，从 a 到 o 要移动 14 位，明文的第二个字母用密钥 14 的加法密码加密.依次类推，密钥字codebreaking表示将明文中的前 12 个字母分别用密钥为 2，14，3，4，1，17，4，0，10，8，13，6 的加法密码加密.然后又重复使用密钥字，将以下的 12 个字母再依次用这些密码加密.如此反复使用，直到整篇文章被全部加密.

练习 2

找一篇稍长一些的英文文章.统计各个英文字母在其中出现的频率（也就是做练习 1）.然后，选定一个密钥字，用上述的维吉尼亚密码加密，再对密文重新统计各个英文字母出现的频率，看这些频率的是否还有显著的差别.

采用多表密码之后，各个字母出现频率的差别被抹杀了.但是，由于不同的密码表的更换是周期性的，仍然给破译者留下了可乘之机.比如，上面的密钥字长度是 12，密码表更换的周期是 12.如果知道了这个周期，即使不知道密钥字，也能用破译单表密码的方法来破译它.方法是：将密文的前 12 个字母排为第一行，以下 12 个字母排成第二行，再下面 12 个字母排成第三行，…，照

此下去，每 12 个字母排成一行，各行上下对齐. 这样，每一列的字母都是用同一张密码表加密，可以按照单表密码的破译方法来破译，(实际上可以按加法密码来破译).

假如预先不知道密钥字的长度，也就是不知道密码表更换的周期，能否先破译出这个周期来呢？可以采用下面的方法：

练习 3

由一个同学选定一篇稍长一些的文章，选定一个密钥字，用维吉尼亚密码加密. 将所得的密文交给另外的同学去破译所用的密钥字的长度.

破译者采用以下方法：对 $n=1,2,3,\cdots$，将密文向左移动 n 位排在原来的密文下面. 这样，第一行的每个密文字母所对准的第二行的字母，恰是第一行的这个字母在密文中向后移动 n 位的那个位置的字母. 依次比较上下两行中处于同一列的字母，统计出上下两个字母相同的列在被统计的列的总数中所占百分比. 观察对不同的 n，这些百分数的变化情况. 是否对某个 n，这个百分数显著上升？这说明什么问题？

设密钥字的长度(也就是密码表的更换周期)为 p. 当 n 不是 p 的倍数时，每一列的上下两个字母是用不同的密码表加密的，相互之间可以认为是无关的，两个字母相同的概率是 $1/26\approx 0.038$. 如果 n 是周期 p 的倍数，则上下两个字母是用同一张表加密的，二者相同的概率可以这样来算：设 26 个字母 a,b,c, ⋯ ,x,y,z 在文章中出现的频率依次是 $k_a,k_b,k_c,\cdots,k_x,k_y,k_z$. 则上下两个字母同为 a 的概率是 k_a^2，同为 b 的概率是 k_b^2，⋯. 于是上下两个字母相同的概率是 $k_a,k_b,\cdots,k_y,k_z$ 的平方和，大约是 0.065，显著增大.(试根据你自己的统计数据计算这个概率.)由这个原理就可以破译出周期 p.

以上所显示的是如何利用统计数据的不均衡性破译密码. 这只是指出一个原理，实际的密码破译当然比这艰苦得多. 但至少你应当获得一个印象，好的密码应当尽量避免各种统计数据的不均衡性.

14.3 现代序列密码体制

古典的密码体制，加密和解密都用人工或机械进行，速度太慢，不能满足快速通讯的需要. 现代的密码体制已是建立在电子技术基础之上. 其中应用很广的一类密码是 0－1 密码体制，即用由 0,1 两个符号组成的序列来作为明文

和密文. 比如，由 0,1 这两个数字组成的二进制五位数共有 32 个，可以表示 26 个英文字母以及空格、标点符号等不同的符号. 于是任何一篇用英文写成的文章就可以用 0－1 序列来代表了. 0－1 序列与英文字符的这种对应可以是公开的，因而表示英文文章的这种 0－1 序列也还是明文. 在保密通讯中，需要将这种以 0－1 序列出现的明文加密成别人读不懂的 0－1 序列，这就是密文. 合法的接收者再将密文翻译回原来的明文.

假设 $A=\{a_i\}=(a_1,a_2,\cdots)$ 是表示明文的 0－1 序列. 为了将它加密成作为密文的 0－1 序列 $C=\{c_i\}$，加密者采用某种方式产生一个 0－1 序列 $B=\{b_i\}$，称为乱数流. 加密者和合法接收者可以用同样的密码机产生同样的乱数流，但要使别人不能产生这个乱数流. 也就是说，乱数流是保密的. 但乱数流与明文无关，它并不带有需要传递的信息，所以它不是密文. 但在需要进行保密通讯传送明文时，加密者就将明文序列 A 与乱数流序列 B 按位进行模 2 的加法得到一个新的 0－1 序列 C 作为密文发出去. 具体来说，$C=\{c_i\}$ 的第 i 位 $c_i=a_i\oplus b_i$ 由 A,B 的第 i 位模 2 加得到. 这里的"模 2 加"是指 c_i 等于 a_i+b_i 除以 2 的余数. 具体来说，加法表为

$$0\oplus 0=0,\quad 0\oplus 1=1,\quad 1\oplus 1=0$$

我们记 $C=A\oplus B$. 合法的接收者收到密文 C 之后，自己产生同样的乱数流序列 B，将 C 与 B 再作模 2 加，得到的 0－1 序列 $C\oplus B$ 就是原来的明文序列 A. 而别的人由于不知道序列 B，也就不能由密文序列 C 得出明文序列 A. 这就达到了保密的目的.

怎样产生用作加密的乱数流序列 B？一个办法是用移位寄存器. 移位寄存器的构造和功能可用示意图表示如下：

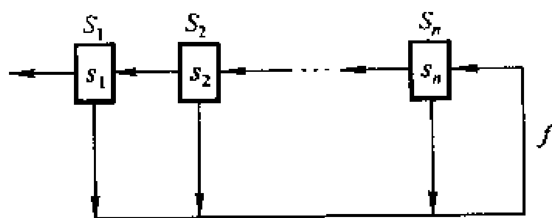


图 14－1

其工作原理是：在 n 个寄存器 $S_1,S_2,\cdots,S_n$ 中存放有 n 个元素 $s_1,s_2,\cdots,s_n$，取值都是 0 或 1. 寄存器中有一个而定的 n 元多项式函数 $y=f(x_1,x_2,\cdots,x_n)$，由自变量 x_i（取值都为 0 或 1）的每一组值得到一个函数值 y（取值仍为 0 或 1）. 寄存器每工作一次，将 S_1 中的 s_1 发出去，S_2 中的 s_2 移进 S_1，S_3 中的 s_3 移进 S_2，…，S_n 中的 s_n 移进 S_{n-1}，而将 $s_{n+1}=f(s_1,s_2,\cdots,s_n)$ 输入 S_n 中，这样就完成了上一个状态 $(s_1,s_2,\cdots,s_n)$ 到下一个状态

$(s_2,s_3,\cdots,s_{n+1})$ 的转移，而发出了一个元素 s_1. 然后再重复这个过程，将 S_1 中的 s_2 发出，状态转移为 $(s_3,s_4,\cdots,s_{n+2})$，其中 $s_{n+2}=f(s_2,s_3,\cdots,s_{n+1})$. 这样不断重复下去，就发出了一个无穷的 0－1 序列 $S=(s_1,s_2,\cdots,)$，其中的递推关系式 $s_{n+i}=f(s_i,s_{i+1},\cdots,s_{i+n-1})$ 对所有的 $i\geqslant 1$ 成立. 这样得到的序列 S 称为移位寄存器序列. 我们希望把它作为乱数流序列来使用. 为此，就希望别人不容易重新把它产生出来. 注意，S 由移位寄存器序列的初态 $(s_1,s_2,\cdots,s_n)$ 和多项式 f 唯一决定. 移位寄存器每发出一个序列元素，状态就转移一次. 而不同的状态(即不同的 n 位 2 进制数)总共只有 2^n 个. 状态转移至多 2^n 次之后就要重复. 而状态一旦重复，序列也就重复. 因此，移位寄存器序列是周期序列，周期至多为 2^n. 为了不让人破译出序列，周期显然应当越长越好，为此就需要增大 n，但 n 受硬件制造条件及成本限制，不可能无限制的提高. 退而求其次，在 n 已经固定之后，希望序列的周期真正达到最大值 2^n. 这就需要选择适当的多项式 f. 周期达到 2^n 的移位寄存器序列称为 M－序列，它的产生和使用比较复杂，这里就不能多涉及了.

如果 f 是线性函数 $f(x_1,\cdots,x_n)=a_1x_1+\cdots+a_nx_n$，(其中系数 $a_1,\cdots,a_n$ 是常数，取值为 0 或1)，这样的移位寄存器称为线性移位寄存器. 此时如果初态 $(s_1,s_2,\cdots,s_n)=(0,0,\cdots,0)$，则以后的状态永远都是 $(0,0,\cdots,0)$，序列全部由 0 组成. 这样的序列显然不能用作加密. 因此，线性移位寄存器序列的状态至多只能有 2^n-1 个，周期至多只有 2^n-1. 达到这个周期的序列称为 m－序列. 注意，如果 $a_1=0$，则序列的递推关系为 $s_{n+1}=a_2s_2+\cdots+a_ns_n$，$s_{n+1}$ 只与 $n-1$ 位的状态 $(s_2,\cdots,s_n)$ 有关. 这样的非零的状态至多只有 $2^{n-1}-1$ 个，序列周期不可能达到 2^n-1. 因此，产生m－序列的线性移位寄存器中的 $a_1=1$. 当然，这个条件只是必要的而不是充分的. 到底怎样的系数集合 $(1,a_2,\cdots,a_n)$ 能够产生m－序列，需要用到近世代数的知识. 不过，你可以用实验方法先体会体会.

练习 4

(1)对 $n=5$. 选取几个不同的线性函数 $f=x_1+a_2x_2+\cdots+a_5x_5$，从任意初态出发，观察所产生的序列的周期是多少？

(2)对 $n=6,7,\cdots,10$，用试验方法各找出一个产生m－序列的线性函数 f.

由古典的单表密码的破译方法知道，密文如果呈现出统计性质的不均衡性，就可能成为被破译方可以利用的弱点. 因此，在使用序列密码时，要求所用

的密文序列好象是一个随机序列.由完全随机的 0－1 随机数组成序列称为随机序列.比如可以用仍硬币来产生：每仍一次硬币，如果正面朝上就记 0，反面朝上就记 0，这样得到的就是随机序列.用作通讯的序列能够传递信息，当然不可能是真正的随机序列.但我们希望它在表面上看起来像是随机序列，具有随机序列的一些主要的性质，让你难以把它和真正的随机序列区别开来.这样的序列称为**伪随机序列**.比如，以下就是随机序列的几个重要性质，被用来检验伪随机序列.

频数检验：在序列中出现的 0 与 1 应当近似地各占一半.

序列检验：序列中相邻两项组成的元素组，00，01，10，11 四种可能应当约各出现 1/4.

扑克检验：序列中相邻 5 项组成的元素组，所有的 2^5 种可能性出现的频率大体相等.

自相关检验：对于每个自然数 p，将序列中 $a_i = a_{i+p}$ 的频率记为 $k(p)$，则 $k(p)$ 应当与 p 无关.

游程检验：游程是指序列中一连串相等的元素系列（同为 0 或同为 1），它们前面或后面的元素与它们都不同.全是 0 的一串元素组成的游程称为 0－游程，全是 1 的一串元素组成的游程称为 1－游程.在随机序列中，长度为 1 的游程占游程总数的 1/2，长度为 2 的游程占游程总数的 1/4，长度为 3 的游程占游程总数的 1/8.一般的，如果游程总数超过 2^{m+1}，则长度为 m 的游程约占游程总数的 $1/(2^m)$.要求伪随机序列也具有同样的性质.

练习 5

用 Mathematica 产生 0 或 1 的随机整数的函数 **Random[Integer]** 产生 0－1 随机序列（实际上它仍是伪随机序列）。再用练习 4 的方法产生一个 m－序列.对这两个序列进行上面所说的各种检验.

m－序列的伪随机性是好的.但由于它是周期序列，所以当 p 等于周期和不等于周期时的自相关函数 $k(p)$ 悬殊很大.另外，m－序列太有规律，并不适合于直接用作实际加密，还要经过加工才能实用.

14.4 公开密钥

在前面介绍的密码体制中，消息的发送者和合法接收者必须预先约定加

密和解密方案,而这一方案不能被别人(特别是敌方)知道.但是,现代社会的信息交流日益频繁,人们常常面临这样的情况:两个预先未联系过的人需要进行保密通讯,信息内容需要向其他人保密.为此,发展出这样的密码体制:每个通讯者向社会公布加密方法,就好象公布自己的电话号码一样;但解密方法则只有自己才知道.任何一个人如果想向通讯者 A 发送消息,只要从公开的"电话号码本"上查出 A 的加密方法,将消息加密后发送给 A.而 A 再用自己独自知道的解密方法将收到的密文译成明文.所有别的人,(包括消息的发送者在内),都不知道 A 的解密方法,这就达到了保密通讯的目的.当然,实际上加密方案和解密方案的原理都是众所周知的,而实现加密和解密除了知道原理外还需要知道所用的参数,称为"密钥".而每个通讯者公布的是加密密钥,而将解密密钥秘而不宣.这样的密码体制称为公开密钥密码体制,简称公钥密码.公钥密码设计的前提是:很难从已经公开的加密密钥破译出解密密钥.

怎样实现公钥密码呢?人们想了各种办法.下面介绍的是 RSA 体制是公钥密码的一种,它的原理主要基于很大的自然数的因子分解的困难.

RSA 公钥体制的原理和方法

每个通讯者秘密选定两个很大的素数 p,q.计算出它们的乘积 $n = pq$.再选定一个与 $(p-1)(q-1)$ 互素的自然数 h.将 n 与 h 作为加密密钥公布出去.如下的加密方法预先大家都知道.消息发送者将自己的明文按某种预先规定好的方式分组,使得每一组可以用 0 到 $n-1$ 之间的一个整数 m_i 来表示.于是明文被表示为 0 到 $n-1$ 之间整数组成的序列 $m_1, m_2, \cdots$.然后他利用消息接收者的公开密钥 n,h 对序列的每一项 m_i 加密成 0 到 $n-1$ 之间的另一整数 $f(m_i)$,得到密文序列 $f(m_1), f(m_2), \cdots$ 发出去.由 m 得到 $f(m)$ 的方法为:将 m^h 除以 n 求余数,得到的余数就是 $f(m)$.

m^h 可能是个很大的数,没有必要先求出它之后再除以 n 求余数.在由低次向高次求 m 的幂的过程中,求出的每个幂都可以用它除以 n 的余数代替.求 m 的幂也不必一次一次乘上去.而可以将 m 的幂一次又一次地平方求得所有的 2^k 次幂,再用这些幂相乘得出 m 的任意次幂.

现在的问题是:消息接收者怎样解密?别人为什么猜不出他的解密方法?

如果两个整数 a,b 除以 n 的余数相同,也就是 $a-b$ 被 n 整除,我们就称 a,b 模 n 同余,记为 $a \equiv b \pmod{n}$.这样,由 m 求 $f(m)$ 的加密方法表达为 $f(m) \equiv m^h \pmod{n}$,可以说 $f(m)$ 是在模 n 的同余类中求 m 的 h 次幂.而解密方法是要反过来由 $f(m)$ 求 m,也就是在模 n 的同余类中求 $f(m)$ 的 h 次方根,这是很难的问题.消息接收者秘密掌握了一个自然数 d 作为解密密钥,满足条件:对任何自然数 a,$(a^h)^d \equiv a \pmod{n}$.有了这个密钥 d,由 $f(m)$ 求 m 的解密方法就变得很简单:$f(m)^d \equiv m \pmod{n}$,只要求 $f(m)$ 的 d 次幂除

以 n 的余数就行了.

这个解密密钥 d 是怎样求出来的? 别人能否由公布的加密密钥 n,h 求出解密密钥 d 来? 这个问题是 RSA 保密性的关键.

消息接收者虽然公布了 $n = pq$ 和 h ,但绝不能公布 p 和 q . 而别人很难将 n 分解得出 p,q ,这是 RSA 体制保密性的关键.(由于现代计算机技术的发展,大数分解问题已有一定的进展,这对 RSA 体制是一个威胁.)根据初等数论的知识,对任何整数 a,k ,成立 $a^{1+k(p-1)(q-1)} \equiv a(\bmod n)$,这称为**欧拉定理**. 这也就是说,任何整数 a 的各次幂除以 n 的余数组成的序列 $a,a^2,\cdots,a^k,\cdots$ 都以 $(p-1)(q-1)$ 为周期,(不一定是最小周期). 由于 h 与 $(p-1)(q-1)$ 互素,同余方程 $hx \equiv 1(\bmod n)$ 有解,而且很容易用辗转相除法求出 0 到 $(p-1)(q-1)-1$ 之间的唯一的解 $x = d$ 来,这个解 d 满足条件 $hd = 1 + k(p-1)(q-1)$,k 是某个正整数. 于是对任意整数 $0 \leqslant m < n$ 有

$$f(m)^d \equiv (m^h)^d \equiv m^{1+k(p-1)(q-1)} \equiv m(\bmod n)$$

这说明了 d 确实是解密密钥.

求解密密钥 d 的关键运算是解同余方程 $hx \equiv 1(\bmod n)$. 在已知 n, h, $(p-1)(q-1)$ 的情况下,这是很容易的. 但现在只公布了 n、h. 别的人不知道(也很难求出) p,q 因而不知道 $(p-1)(q-1)$, 就求不出 d 来. 由于 $(p-1)(q-1)$ 很大,也不可能用穷举法求 a^k ($k = 1, 2, 3,\cdots$)模 n 的余数组成的序列的周期从而求出 $(p-1)(q-1)$. 这就达到了保密的目的.

练习 6

(1)选定一个素数 p (不必太大). 列出一个表,每个底数 $a = 1,2,3,\cdots$ 占一行,每个指数 k 占一列,在第 a 行第 k 列填写 a^k 除以 p 的余数. 观察每一行的循环周期以及最小的公共周期. 哪些行达到了最大周期?

对不同的 p 作实验,看 $p-1$ 是否总是各行的最小的公共周期.

选定两个素数 p,q (不必太大). 列出一个矩形表,使表中第 a 行第 k 列的元素为 a^k 除以 pq 的余数. 观察各行的循环周期以及最小的公共周期.

用不同的 p,q 作实验,看 $(p-1)(q-1)$ 是否总是各行最小的公共周期.

(2)选定大素数 p,q ,以及与 $b = (p-1)(q-1)$ 互素的正整数 $h < b$. 用实验二第 3.5 中所说的方法求不定方程 $hx - by = 1$ 的最小正整数解 (x,y). 则 x 是同余方程 $hx \equiv 1(\bmod b)$ 的最小正整数解,必然小于 b ,可以充当解密密钥.

(3)自己选定大素数 p,q 及 h ,计算出解密密钥 d . 向你的同伴公布 $n = pq$ 及 h ,让他利用 RSA 算法向你发送加密信息,你进行解密. 看他能否破译

你的解密方法.(由于你所采用的 p,q 不是很大,他有可能破译你的解密密钥.)

实验十五　初等几何定理的机器证明

从 17 世纪初开始，人类历史上经历了一场史无前例的技术革命，以蒸汽机为代表的机器代替了各种类型的体力劳动. 这场革命实现了体力劳动的机械化，从而极大地促进了社会生产力的发展. 本世纪计算机技术的迅猛发展又为人类脑力劳动的机械化创造了物质条件. 在一些领域，计算机已部分代替了人类的脑力劳动. 其中最引人注目的是计算机自动推理能力的突破性进展. 1998 年，IBM 公司的大型计算机深蓝战胜了国际象棋冠军卡斯帕罗夫. 这一轰动一时的事件被国际新闻媒体评为当年的十大新闻之一. 深蓝战胜卡斯帕罗夫带来的震撼不仅只限于体育界及计算机学界，因为它给人类提出了这样的一个问题：计算机的智力是否可以达到或超过人类本身的智能？当然，现在担心这个问题未免有点杞人忧天，也许计算机永远无法达到人类的智力水平. 然而，让计算机部分代替人脑实现人力难以胜任的繁琐的计算与推理，使人类自身致力于更富创造性的劳动是非常迫切与重要的. 正是基于这一认识，我国著名的数学家吴文俊院士从 70 年代开始致力于数学机械化(即数学定理与结论的计算机自动推导)的研究，并取得了巨大的研究成果，在国际上获得了崇高的声誉. 他提出的用于求解多项式方程组的解法被国际学术界命名为“吴方法”.

所谓数学定理的机械化证明，即把某一类定理作为一个整体加以考虑，建立一个统一的、确定的证明程序，对于该类定理中的每一个定理，应用证明程序进行有限步推理之后即可从数学命题的假设推断出数学命题的结论在若干附加条件下是否为真. 实现数学定理证明的机械化是数学学科发展中的一次重大飞跃.

本实验将以初等几何定理的机器证明为例，让学生理解计算机是如何实现自动推理的，并使学生认识机械化的数学思想在数学发展中的重要地位.

15.1　几何问题代数化

当代电子计算机的主要功能是数值或符号的运算与推理. 因此，要实现几何定理的机器证明的首要步骤是将几何问题转化为代数问题. 10 世纪伟大的数学家笛卡儿创立的解析几何为解决这一问题提供了思想与理论基础. 解析

几何的基本思想是，依据不同的几何对象，引进适当的坐标系，这样，几何对象以及它们之间的关系就可以用代数关系式描述，几何问题也因此可以转化为代数问题求解. 笛卡儿有一段名言：

一切问题都可以化为数学问题

一切数学问题都可以化为代数问题

一切代数问题都可以化为代数方程求解

当然，笛卡儿的话未免说得有点绝对，但它表明了代数方程求解的重要性. 实际上，平面几何定理的机器证明正是将几何定理转化为代数方程之间的关系判别问题.

在平面几何中，常见的几何形体有：点，线段，角，三角形，四边形，圆，椭圆，抛物线，双曲线等. 在建立了恰当的坐标系后，这些几何形体可以用一对坐标，或一个线性方程或一个二次方程来表示. 几何形体之间的关系常见的有：三点共线，点在线上，三线共点，两线平行，两角相等，两三角形相似，线与圆相切，两圆相切等等. 这些关系也可以转化为多项式方程. 下面，让我们看一个具体的实例.

例 1　证明：平行四边形的两条对角线互相平分.

设平行四边形 $ABDC$ 的两条对角线 AD 与 BC 相交于 N（如图 15－1 所示），我们要证 $AN = ND, BN = NC$.

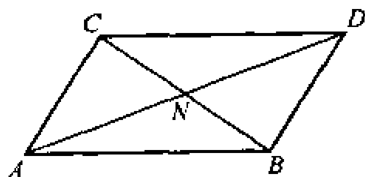


图 15－1

为将上述命题转化为代数问题，我们建立如下的直角坐标系. 取 A 为原点，AB 为 x 轴的正向. 则 A,B,C 点的坐标可设为：$A = (0,0)$，$B = (u_1,0)$，$C = (u_2,u_3)$. 其中，u_1,u_2,u_3 为自由参数. 再设 $D = (x_1,x_2)$，$N = (x_3,x_4)$. 如果 A,B,C 确定了，则 D 与 N 也可确定. 因此，x_1,x_2,x_3,x_4 由 u_1,u_2,u_3 所确定.

由条件 $AB \parallel CD$ 及 $AC \parallel BD$，我们有

$$0 = \frac{x_2 - u_3}{x_1 - u_2}$$

$$\frac{u_3}{u_2} = \frac{x_2}{x_1 - u_1}$$

消去上式中的分母，我们得到多项式方程

$$h_1 = x_2 - u_3 = 0 \tag{1}$$

$$h_2 = (x_1 - u_1)u_3 - x_2u_2 = 0 \tag{2}$$

再由 A,N,D 共线以及 B,N,C 共线得到

$$\frac{x_4}{x_3} = \frac{u_3}{x_1}$$

$$\frac{x_4}{x_3 - u_1} = \frac{u_3}{u_2 - u_1}$$

消去分母得

$$h_3 = x_4x_1 - x_3u_3 = 0 \tag{3}$$

$$h_4 = x_4(u_2 - u_1) - (x_3 - u_1)u_3 = 0 \tag{4}$$

这样，定理的假设转化为上述四个代数关系式.

由 $AN = ND$ 及 $BN = NC$，我们可以将结论转化为下列两个方程

$$g_1 = x_1^2 - 2x_1x_3 - 2x_4x_2 + x_2^2 = 0 \tag{5}$$

$$g_2 = 2x_3u_1 - 2x_3u_2 - 2x_4u_3 - u_1^2 + u_2^2 + u_3^2 = 0 \tag{6}$$

于是，原定理转化为如下代数问题：假设(1)—(4)成立，则(5)与(6)也成立.

显然，几何定理的代数转化方式与结果并不唯一. 如在上述实例中，利用向量 AD 等于向量 AB 加向量 AC，易知点的 D 坐标为 $D = (x_1,x_2) = (u_1 + u_2,u_3)$. 于是我们有条件

$$h_1' = x_1 - u_1 - u_2 = 0 \tag{7}$$

$$h_2' = x_2 - u_3 = 0 \tag{8}$$

上述条件显然比条件(1)与(2)更简单. 实际上，利用(7)及(8)可以消去变元 x_1,x_2，从而进一步简化条件与结论.

练习 1

重新考虑例 1. 仍然设 A 点为原点，但 D 点的坐标 $D = (u_1,u_2)$ 任意选取(因而坐标系随 D 点的坐标而变化).

(1) B 点的坐标有几个是自由的？如何设定 B 点的坐标？

(2)根据以上选取，将例 1 中定理的条件与结论转化为代数方程.

练习 2

将下列几何叙述“翻译”成一个或几个多项式方程.

(i)　A,B,C 三点共线.

(ii)　线段 AB 与 CD 相交于 E.

(iii)　从 A 向 BC 引垂线，D 为垂足.

(iv) AB,CD,EF 三线相交于一点.

(v) A,B,C,D 四点共圆.

(vi) BD 平分角 ABC.

(vii) AB 与过 A,C,D 的圆相切.

练习 3

将下列定理中的条件与结论转化为多项式方程.

(i) **Apollonius 圆定理** 设三角形 ABC 是直角三角形，A 为直角顶点. 则三角形三边的中点以及 A 在 BC 边上的垂足共圆.

(ii) **Simson 定理** P 设为三角形 ABC 外接圆上一点. 从 P 点向三角形 ABC 三边 AB,BC,CA 引垂线，垂足分别为 L,M,N. 求证：三垂足 L,M,N 共线.

(iii) 给定平面上的三角形 ABC. 分别以 AB,BC,CA 为边做等边三角形 $\triangle A'BC$，$\triangle AB'C$，$\triangle ABC'$. 则 AA'，BB'，CC' 相交于一点.

通过引进坐标系，我们可以将几何定理转化为代数关系式. 但是，几何定理与代数关系是否完全等价？让我们先做以下练习.

练习 4

再次考察例 1. 取 $(u_1,u_2,u_3,x_1,x_2,x_3,x_4)=(2,1,0,2,0,1,0)$. 验证上述取值是否满足(1)—(4)？是否满足(5)及(6)？由此回答：由(1)—(4)成立是否可以推出(5)及(6)成立？例 1 中的定理是否与转化后的代数问题等价？你如何解释这种现象？如何将几何定理转化为等价的代数问题？进一步对练习 2 与练习 3 中的几何定理(成叙述)与其代数表达式的等价问题做类似的考察，其中要特别注意练习 3 中的问题(iii).

上述练习告诉我们，在将几何问题转化为代数问题时要十分小心. 一般来说，转化后的代数问题只有在某些条件下才与原几何问题等价.

15.2 代数关系式的推导与验证

从上一节可以看到，要实现几何定理的机器证明，关键是要对以下问题给出系统的、程序化的求解方案：

问题　给定多项式方程组：

$$\begin{cases} h_1(u_1,\cdots,u_m,x_1,\cdots,x_n)=0 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ h_n(u_1,\cdots,u_m,x_1,\cdots,x_n)=0 \end{cases} \tag{9}$$

其中 $u_1,\cdots,u_m$ 为自由参数.则在某些条件下,可以由上述方程组推导出

$$g(u_1,\cdots,u_m,x_1,\cdots,x_n)=0 \tag{10}$$

(由于多个结论同一个结论是类似的,我们不妨设结论中只有一个方程).

如果以 $u_1,\cdots,u_m$ 为参数能从方程组(9)中求解出 $x_1,\cdots,x_n$,则可以将求得的值代入(10)来验证(10)式.

练习 5

从方程组(1)—(4)中求解出 $x_1,\cdots,x_4$,然后验证(5)及(6)式,从而证明例1中的定理.该定理成立的条件是什么?你能否给出定理成立条件的几何解释?

除了极其简单的情况,要想从方程组(9)中求解出 $x_1,\cdots,x_n$ 是不切实际的(读者只需考虑练习3中的几个问题即可).为了求解问题,需要更加有效的途径.

回忆在单变量的情形,如果想从 $h(x)=0$ 推出 $g(x)=0$,我们并不需要求出 $h(x)=0$ 的零点.实际上,只需证明 $h(x)$ 是 $g(x)$ 的因子,即存在多项式 $q(x)$ 使得 $g(x)=q(x)h(x)$.对于具有多个变量的问题,我们如能证明存在函数 $q_i(u_1,\cdots,u_m,x_1,\cdots,x_n)$, $i=1,\cdots,n$ 使得

$$g=q_1h_1+\cdots+q_nh_n \tag{11}$$

则显然可以由(9)式推导出(10)式.当然,函数 q_i 一般不再是多项式而是有理式.

我国著名数学家吴文俊先生给出了确定关系式(11)的具体的构造性的算法,这一方法在国际上被誉为"吴方法".吴方法主要由下列几大步骤构成:

- 将方程组(9)化成"三角阵"的形式:

$$\begin{cases} f_1(u_1,\cdots,u_m,x_1)=0 \\ f_2(u_1,\cdots,u_m,x_1,x_2)=0 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ f_n(u_1,\cdots,u_m,x_1,\cdots,x_n)=0 \end{cases} \tag{12}$$

• 计算 g 被 $f_n,\cdots,f_1$ 整除所得的余式：

$$\begin{cases}R_{n-1}=\mathrm{Rem}(g,f_n,x_n)\\R_{n-2}=\mathrm{Rem}(R_{n-1},f_{n-1},x_{n-1})\\\cdots\cdots\cdots\cdots\\R_0=\mathrm{Rem}(R_1,f_1,x_1)\end{cases}\tag{13}$$

其中 $\mathrm{Rem}(R_i,f_i,x_i)$ 表示以 x_i 为主变元用 f_i 整除 R_i 所得的余式.

• 如果 $R_0=0$，则(11)成立.

上述步骤中的一个核心算法是多项式的**伪除法**. 假设 f,g 是两个关于 $x_1,\cdots,x_n,y$ 的多项式，我们以 y 为主变元，将 f,g 写成

$$\begin{cases}f=c_my^m+\cdots+c_1y+c_0\\g=d_ny^n+\cdots+d_1y+d_0\end{cases}$$

其中 c_i,d_i 均是 $x_1,\cdots,x_n$ 的多项式. 设 $n\leqslant m$，我们以 y 为变元做 f 除以 g 的带余除法，则

$$f=qg+r$$

其中 q,r 都是关于 $x_1,\cdots,x_n,y$ 的有理式，它们的分母都是 d_n 的某个次幂(设为 s). 将上式两边同乘 d_n^s，则存在关于 $x_1,\cdots,x_n,y$ 的多项式 $\tilde{q},\tilde{r}$ 使得

$$d_n^sf=\tilde{q}g+\tilde{r}$$

此即所谓的多元多项式的伪除法. $\tilde{r}$ 是 f 除以 g 的余式，记为 $\tilde{r}=\mathrm{Rem}(f,g,y)$. 例如，以 y 为主变元，$f=x^2y^3-y$ 被 $g=x^3y-2$ 的伪除法是：

$$(x^3)^3f=(x^8y^2+2x^5y+4x^2-x^6)g+8x^2-2x^6$$

余式 $\mathrm{Rem}(f,g,y)=8x^2-2x^6$.

下面，我们对吴方法的实现步骤做一个简单的解释.

第一步实际上完全类似于线性方程组的消元. 首先，将方程组(9)的 $n-1$ 个方程中的变元 x_n 消去. 这一过程可以通过多项式之间的伪除法得到. 例如，为将方程组

$$\begin{cases}h_1=x_1^2+x_2^2\\h_2=1+x_1+x_2^3\end{cases}$$

中的一个方程的变元 x_2 消去，我们将 h_2 除以 h_1 得到 $h_3=\mathrm{Rem}(h_2,h_1,x_2)=1+x_1-x_1^2x_2$. 再将 h_1 除以 h_3 得 $h_4=\mathrm{Rem}(h_1,h_3,x_2)=1+2x_1+x_1^2+x_1^6$. 这样，我们将其中一个方程中的变元 x_2 消去了.

在消去变元 x_n 后，再由消去了 x_n 的 $n-1$ 个方程出发，将其中 $n-2$ 个方程的变元 x_{n-1} 消去. 这个过程一直进行到最后即可将原方程组(9)化为三角阵(12)的形式.

吴方法的第二步是依次对 f_i，$i=n,\cdots,1$ 做伪除法. 假设 f_i 关于主变元 x_i 的首项系数为 d_i，$i=1,\cdots,n$. 则

$$\begin{cases} d_n^{s_n}g = q_n f_n + R_{n-1} \\ d_{n-1}^{s_{n-1}}R_{n-1} = q_{n-1}f_{n-1} + R_{n-2} \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ d_1^{s_1}R_1 = q_1 f_1 + R_0 \end{cases} \tag{14}$$

其中 s_i 为非负整数. 从上式可以推出，存在多项式 A_i 使得

$$d_1^{s_1}\cdots d_n^{s_n}g = A_1 f_1 + \cdots + A_n f_n + R_0 \tag{15}$$

第三步可以由(15)立即得出. 但必须注意(11)成立的条件是 d_i，$i=1,\cdots,n$ 不为零.

例 2 我们继续来考察例 1 中的定理. 我们已将该定理的条件与结论化为代数方程组(7)—(8)，(3)—(4)以及(5)—(6). 为了证明该定理，我们需要将条件方程组化为三角形方程组. 首先，我们从(3)—(4)中消去变元 x_4 得

$$f_3 = \mathrm{Rem}(h_3,h_4,x_4) = -u_1u_3x_1 + u_1u_3x_3 - u_2u_3x_3 + u_3x_1x_3$$

由于 h_1',h_2' 已自动对角化，故原条件化成了对角形式

$$\begin{cases} f_1 = x_1 - u_1 - u_2 = 0 \\ f_2 = x_2 - u_3 = 0 \\ f_3 = -u_1u_3x_1 + u_1u_3x_3 - u_2u_3x_3 + u_3x_1x_3 \\ f_4 = x_4(u_2-u_1) - (x_3-u_1)u_3 \end{cases}$$

下面，我们验证结论(5).

$$\begin{aligned} R_3 &= \mathrm{Rem}(g_1,f_4,x_4) = -u_1x_1^2 + u_2x_1^2 + 2u_1u_3x_2 - u_1x_2^2 \\ &\qquad + u_2x_2^2 + 2u_1x_1x_3 - 2u_2x_1x_3 - 2u_3x_2x_3 \\ R_2 &= \mathrm{Rem}(R_3,f_3,x_3) = (u_1-u_2)u_3(u_1x_1^2 + u_2x_1^2 - x_1^3 \\ &\qquad + 2u_1u_3x_2 - u_1x_2^2 + u_2x_2^2 - x_1x_2^2) \\ R_1 &= \mathrm{Rem}(R_2,f_2,x_2) = (u_1-u_2)u_3(u_1+u_2-x_1)(u_3^2+x_1^2) \\ R_0 &= \mathrm{Rem}(R_1,f_1,x_1) = 0 \end{aligned}$$

即结论(5)成立. 类似可验证结论(6)成立.

练习 6

用吴方法验证结论(6)成立. 同时检查结论(5)与(6)成立的条件，并给出相应的几何解释. 该条件与练习 5 中所得到的条件是否一致？

练习 7

用吴方法证明 Apollonius 圆定理与 Simson 定理，并给出定理成立的条件及其几何解释.

练习 8

用吴方法证明练习 3 中的问题(iii). 你成功了吗？如果没有成功，请检查一下原因，并考虑有没有办法去克服其中的困难？

吴方法不仅可以用来证明初等几何定理，而且可以应用于自动推理领域，即由变量之间的已知关系推导出变量之间的未知关系.

例 3 试推导正五边形的边长与连接不相邻的两顶点的线段长之间的关系.

假设正五边形 $ABCDE$ 的中心为 O. 以 O 为原点，平行于 DC 的直线为横轴建立直角坐标系(如图 15－2 所示). 设各点坐标如下：$O(0,0)$，$A(0,x_5)$，$B(x_1,x_3)$，$C(x_2,x_4)$，$D(-x_2,x_4)$，$E(-x_1,x_3)$. 则线段 CD 的长度为 $2x_2$，线段 BE 的长度为 $2x_1$. 所以，我们只要推导出 x_1，x_2 所满足的关系.

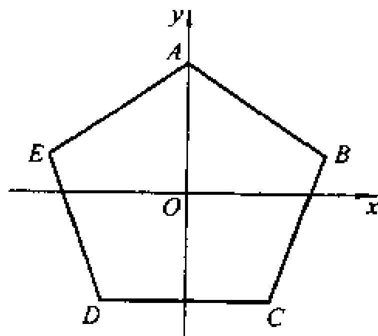


图 15－2

由 $ABCDE$ 为正五边形，我们有

$$\begin{cases} h_1 = x_1^2 + x_3^2 - x_5^2 = 0, & \text{由 } BO = AO \\ h_2 = x_2^2 + x_4^2 - x_5^2 = 0, & \text{由 } CO = AO \\ h_3 = x_1^2 + (x_3 - x_5)^2 - (x_2 - x_1)^2 - (x_3 - x_4)^2 = 0, & \text{由 } AB = BC \\ h_4 = x_1^2 + (x_3 - x_5)^2 - 4x_2^2 = 0, & \text{由 } AB = CD \end{cases}$$

将上述方程组化成三角形方程组得

$$\begin{cases} f_1 = 4x_2^3(x_1 + x_2)(-x_1^2 + x_1x_2 + x_2^2) \\ f_2 = -x_1^4 + 4x_1^2x_2^2 - 4x_2^4 - x_1^2x_3^2 + 4x_2^2x_3^2 \\ f_3 = -2x_1^2 + 2x_1x_2 + 4x_2^2 - 2x_3^2 + 2x_3x_4 \\ f_4 = 2x_1^2 - 4x_2^2 + 2x_3^2 - 2x_3x_5 \end{cases}$$

其中第一个方程即给出了 x_1 与 x_2 之间的关系：

$$-x_1^2 + x_1x_2 + x_2^2 = 0$$

练习 9

试用吴方法推导三角形的面积与三边长之间的关系.进一步,你能否推出四面体的体积与各棱边长之间的关系？

数学机械化的思想及吴方法除了在上述领域有成功的应用之外,还在纯数学领域的各个分支(如构造性代数几何,构造性微分几何,微分方程求解等),理论物理,机械构造学,机器人运动学,计算机视觉,计算机辅助设计,图形图象处理等广泛的领域具有重大应用价值.

附录 Mathematica 程序

1. 例 2 中利用吴方法由(7)—(8)式及(3)—(4)式推导(5)式的程序.

```
h1=x1-u1-u2;
h2=x2-u3;
h3=x4*x1-x3*u3;
h4=x4*(u2-u1)-(x3-u1)*u3;
g1=x1^2-2*x1*x3-2*x4*x2+x2^2;

f1=h1
f2=h2
f3=PolynomialRemainder[(u2-u1)h3,h4,x4]
f4=h4
```

```
R3=Simplify[PolynomialRemainder[(u2-u1)*g1,f4,x4]]
R2=Simplify[PolynomialRemainder[(u1*u3-u2*u3+u3*x1)
                                  *R3,f3,x3]]
R1=PolynomialRemainder[R2,f2,x2]
R0=PolynomialRemainder[R1,f1,x1]
```

2. 例 3 中推导 x_1 与 x_2 的关系的程序.

```
h1=x1^2+x3^2-x5^2;
h2=x2^2+x4^2-x5^2;
h3=x1^2+(x3-x5)^2-(x2-x1)^2-(x3-x4)^2;
h4=x1^2+(x3-x5)^2-4x2^2;

h2=PolynomialRemainder[h2,h1,x5]
h3=PolynomialRemainder[h3,h1,x5]
h4=PolynomialRemainder[h4,h1,x5]
h1=PolynomialRemainder[x3^2*h1,h4,x5]
h3=PolynomialRemainder[h3,h4,x5]
h3=PolynomialRemainder[h3,h2,x4]
h2=PolynomialRemainder[x3^2*h2,h3,x4]
h2=Simplify[PolynomialRemainder[(4x2^2-x1^2)
                                  *h2,h1,x3]]
temp=h1;h1=h2;h2=temp;

Factor[h1]
Factor[h2]
Factor[h3]
Factor[h4]
```

参考文献

1. 吴文俊. 吴文俊论数学机械化. 山东：山东教育出版社，1996

2. 石赫. 机械化数学引论. 湖南:湖南教育出版社,1998

3. 杨路,张景中,侯晓荣. 非线性代数方程组与机器证明. 上海:上海科技教育出版社,1996

Mathematica 简介

● 数值计算与符号计算

数值计算的结果是一个数值，符号计算是相对数值计算而得其名，符号计算的计算结果是一个表达式，数值是表达式的一种简单形式. 例如，因式分解的计算结果仍是多项式，不定积分的计算结果仍是初等函数. 象 PASCAL 和 C 等高级语言，主要用于数值计算，用于计算各类表达式的符号计算系统是一个数学软件系统，它主要包括：数值计算、符号计算、图形功能和程序设计四个方面功能. 它含有功能强大种类丰富的内部函数，用户也可以自由的定义自己的函数并扩充到系统函数中.

符号计算已有几十个软件系统，一般有专用系统和通用系统两类，都有适合于从工作站到微机使用的多种版本. 目前，通用符号计算系统有：(按字母顺序排列 Axiom、Macsyma、Maple、Mathematica、Matlab、Reduce 和 Dirive 等.

符号计算系统通常都有两种运行方式：一种是交互式，每发一个命令，就执行一种相应的数学计算，例如，计算定积分 $\int_{-1}^{1} x^2(11 - \sin x)\mathrm{d}x$ 的函数命令是：

```
Integrate[x^2(11-Sin[x]),{x,-1,1}]
```

另一种方式是写一段程序，系统连续执行一系列的命令，就像用 Fortran 或 C 写程序一样. 每个符号计算系统都有自己的程序设计语言，这些语言与通用的高级语言大同小异.

Mathematica 是优秀的符号计算系统之一. 下面简要列出 Mathematica 系统的主要函数，供使用参考. 要进一步掌握和了解 Matbematica 的功能，请查阅系统的在线帮助和参考有关的书籍.

● 进入 Mathematica

在 windows 3. x 版中，在装好 Mathematica 2. 2 的窗口中双击 Mathematica 的图标 . 在 Win 95 下，在"开始"菜单的"程序"中单击"Mathematica 2. 2". 屏幕显示一个启动图面. 稍停片刻，系统装好内核后，则显示如图 16—1 的工作屏幕.

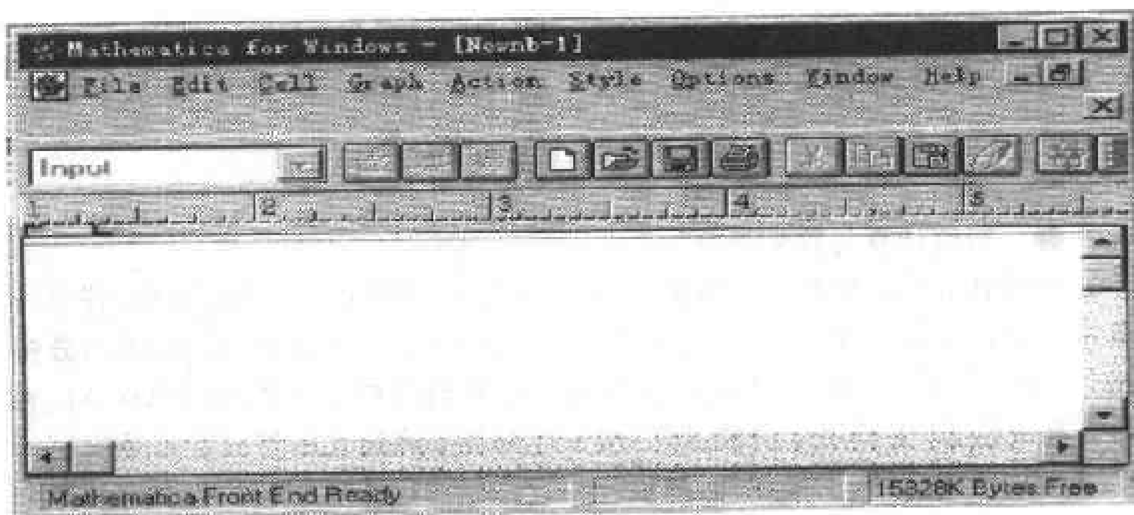


图 16－1 Mathematica 的工作屏幕

Mathematica 中共有 9 个主菜单和一些个工具按钮，菜单意义文如其名. 例如：File 表示文件，Edit 表示编辑，Cell 表示对 Cell(单元)的处理大部分工具按钮也是“图”人、如文义，例如，用剪刀表示剪切. 下列几个常用的工具按钮：



创建、打开、保存、打印 Notebooks 按钮

剪切、复制、粘贴、取消按钮

计算表达式、动画、图形的 Postscript、播放声音

3 维图形视点、颜色选择、中断、帮助按钮

图 16－2 常用工具按钮

● 输入和计算表达式

在 Windows 环境下，工作屏幕象一张长长的草稿纸，称工作屏幕为 Notebook，它是后缀是. ma 的文件. 象处理其它计算机文件一样，我们可以对 Notebook 进行创建、打开、保存、修改和打印等文件操作.

在 Dos 界面下，屏幕上先显示输入计算的标记，“In[1]：=”，用户在输入一个表达式后，按回车键，系统立即开始计算输入的表达式.

在 Windows 环境下，用户先输入一行或多行表达式，再发计算表达式命令(单击运行按钮或按 Shift＋回车键)，系统完成计算后输出计算结果时，再显示输入的行标记“In[1]：=”. 需要注意的是，在 Windows 环境中回车键仅

表示命令之间的分隔标记.

计算表达式的命令方式有:

1)单击运行按钮 ;

2)按 Shift+回车键;

3)按键盘右边的数字键"5"(在非数字状态下).

例如:启动 Mathematica 后,输入

Plot[x^3+3x−1,{x,−2,2}]

单击运行按钮 ,屏幕显示图 16−3.

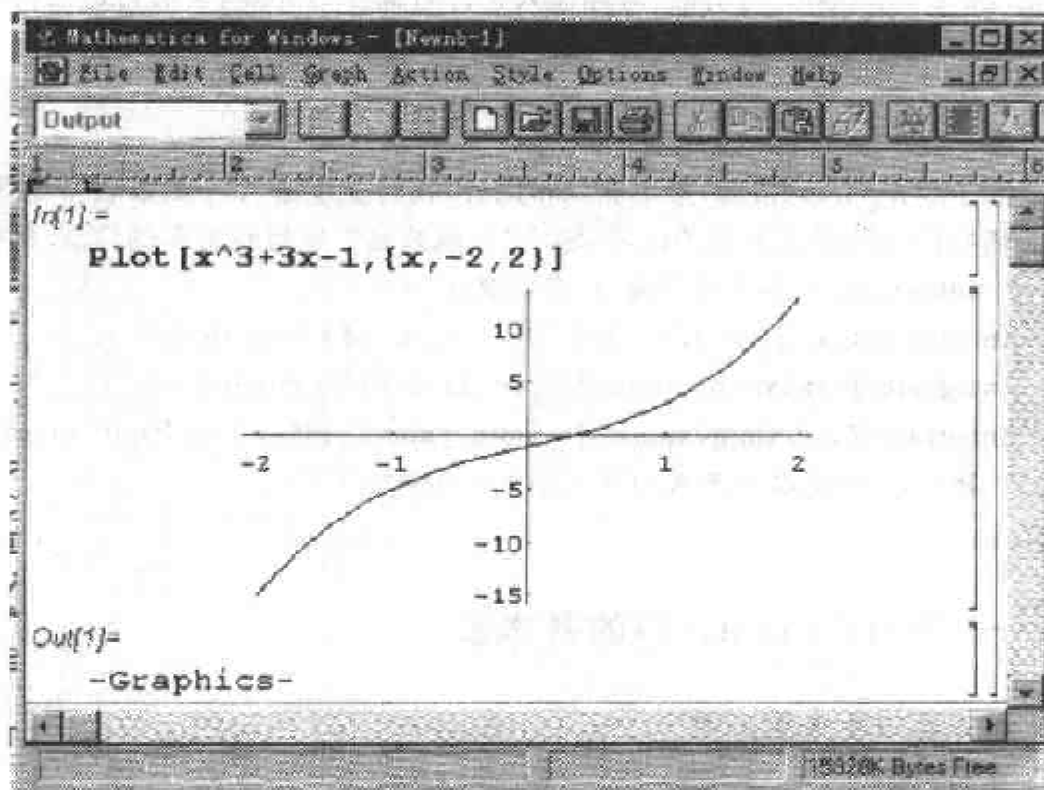


图 16−3 执行一条命令后的窗口

为了紧缩版面,后面的例题中将同行显示"In[x]:="和输入内容,请不要将"In[x]:="误认为是输入的字符.另外,用注释语句(* 楷体 *)形式做例题的简要说明.

● 输入函数或命令的快捷方式

键入函数或命令的第一个或前几个字符，按快捷键 Ctrl+K；或在"Action"菜单中选择子菜单"Prepare Input"的"Complete Selection"命令. 这时屏幕上显示以键入字符为首的函数和命令列表，在列表上单击某个函数或命令，系统则将这条函数或命令粘贴到你的输入中. 单击函数和命令列表框右边的上下箭头，可调出更多的函数和命令. 用户定义的函数也自动放在列表中.

● 退出 Mathematica

要退出 Mathematica，在"文件"菜单中选择"Exit"命令或按"Alt+F4". 如果窗口中还有内容没有保存到 Notebook 中，这时会出现一个对话框，系统问你是否保存，单击对话框上的"否(N)"按钮则关闭窗口；单击"是(Y)"按钮，调出"SaveNotebook"对话框，等你输入文件名保存 Notebook 后再退出窗口. 退出后返回到调入 Mathematica 以前的状态.

● 在 Mathematica 中获取帮助

除了单击 Help(帮助)菜单获取帮助外，在行文中输入"? 函数名"，可得到有关函数的调用形式和相关说明，输入"??函数名"，则得到更多的信息. 例如：

? **Integrate**(* 单击运行按钮，系统显示如下 *)

Integrate[**f**,**x**] gives the indefinite integral of f with respect to x

Integrate[**f**,{**x**,**xmin**,**xmax**}] gives the definite integral

Integrate[**f**,{**x**,**xmin**,**xmax**},{**y**,**ymin**,**ymax**}] gives a multiple integral

? **P***(* 列出以 P 开头的所有函数和实体 *)

一、Mathematica 中的基本量

● 数与数的表示

数值类型

Mathematica 的简单数值类型有整数、有理数、实数和复数. 整数(Integer)由若干个 0 到 9 的数字组成，数字之间不能有空格、逗号和其它字符，正负号放在整数的首位，输入时，正号可省略不写，只要内存允许，Mathematica 可以表示任意长度的精确整数，不受所用的计算机的字长所限制. 在 Mathematica 中允许使用分数，也就是用有理数(Rational)表示一个化简过的

分数.当两个整数相除而又不能整除时,系统就用有理数来表示.例如:$\frac{2}{91}$,系统自动对分数做可能的化简,自动约去分子和分母的公因子.输入时,我们用分子/分母的形式,用除号隔开分子与分母,输出时,系统一般用分式的形式,当分子和分母数值较大时,有时也以分子/分母的形式表示.

类型	描述	例	特征说明
整数	**Integer**	1234567	任意长度的精确整数
有理数	**Rational**	13243/5768971	化简过的分数
实数	**Real**	1234567.0	任意精确度的近似实数
复数	**Complex**	1234567.0+3.2I	实部和虚部可为整数、有理数、实数

数学常数

数学常数	意义
Pi	π=3.141 59
E	自然对数的底 e=2.718 28…
Degree	1 度,π/180
I	虚数单位 $i=\sqrt{-1}$
Infinity	∞,无穷大

这些数学常数都是精确数,用在公式推导和计算中,在公式推导中数学常数多为精确数;用在数值计算中的数学常数可以取任意精确度.

数的输出

N[表达式]	以实数形式输出表达式
N[表达式,n]	以 n 位精度的实数形式表示表达式

例如,**N[Pi,30]**,取 30 位数字的 π.

● 变量

给变量取名

在 Mathematica 中内部函数或命令都是以大写字母开头的标识符,为了避免混淆,在 Mathematica 中变量名通常以小写英文字母开头,后跟字母或数字,变量名字符的长度不限.例如:abcdefghijk,A,x3 都是合法的变量名;而 2t 和 u v(u 与 v 之间有一空格)不能作为变量名.在变量名中英文字母大小写意义不同,因此 A 与 a 表示两个不同的变量.在 Mathematica 中变量即取即用,不需要先说明变量的类型再使用.

给变量赋值

在 Mathematica 中用"="或":="起赋值作用,一般形式为:

变量=表达式　或　变量 1=变量 2=表达式

变量=.　　清除变量的值

clear[变量]　　清除变量的定义和值

在 Mathematica 中"="应理解为给变量定义一个值. 在 Mathematica 中，一个变量可以表示一个数值、一个表达式、一个数组或一个图形.

变量替换

变量有两种获取数值的方式，定义其值是一种方法，另一种是对变量做替换.

expr/. lks->rhs　用 **rhs** 替换 **expr** 中的 **lhs**

expr/. {lhs1->rhs1, lhs1->rhs1, …}　分别用 **rhsi** 替换 **expr** 中的 **lhsi**

例如：

In[1]:=**f=x^2+1;f/. x->1**(*用 1 替换 f 中的 x*)

Out[1]=2

In[2]:=**g=x^2+y^2;g/. {x->2, y->1}**(*在 g 中替换 x, y 两个变量*)

Out[2]=5

● 表

循环描述

表在形式上是用花括号括起来的若干元素，元素之间用逗号分隔. 在运算中，可对表作整体操作，也可对单个元素操作. 表可以表示数学中的集合、向量和矩阵，也可表示数据库中的一组记录. 表中的元素可为任何数据类型. 表的元素可为数值、表达式和表. 同一个表中的元素可为不同的数据类型. 表中元素较少时，直接使用形式：变量={元素 1，元素 2，…}键入表中元素，在给出表名的同时又建立了表中元素. 更多的时候需要用建表命令. 在学习建表命令前，先了解 Mathematica 中的循环描述.

循环描述形式：{循环变量，循环初值，循环终值，步长}

循环变量、循环初值、循环终值和步长可为整数、有理数和实数. 当步长为 1 时可省略，当循环初值为 1 时可省略. 例如：

{i, max}i　从 **1** 到 **max**；　　**{max}**　重复 **max** 次；

{i, imin, imax}, {j, jmin, jmax}　　**i** 从 **imin** 到 **imax**，并对 **i** 的每一个值，**j** 从 **jmin** 到 **jmax**

建表函数

Table[通项公式，{循环范围}，{循环范围}]

按循环范围定义表，用通项公式的规则定义表的元素，循环范围可为一重至多重.

Range[初值,终值,步长]

生成值为{初值,初值+步长,..,终值}的数值表

In[1]:=**Range**[**3**,**9**,**2**]

Out[1]={3,5,7,9}

In[2]:=**b**=**Table**[**i**+**j**−**1**,{**i**,**1**,**3**},{**j**,**1**,**3**}]

Out[2]={{1,2,3},{2,3,4},{3,4,5}}

表的分量表示

在上例中,当 **b** 表示一个表时,**b**[[**2**]]表示 **b** 中的第 **2** 个子表,即第二行元素.如果 **s**={**1**,**3**,**5**,**7**},那 **s**[[**2**]]的值是 **3**;下列表示表的分量的一般形式和实例.

表的分量	意义
t[[**n**]]或 **Part**[**t**,**n**]	**t** 中的第 **n** 个子表
t[[−**n**]]或 **Part**[**t**,−**n**]	倒数第 **n** 个元素
First[**list**]	**list** 中的第一个元素
Last[**list**]	**list** 中的最后一个元素
t[[{**n1**,**n2**,…}]]或 **Part**[**t**,{**n1**,**n2**,**n3**,…}]	给出由 **t** 的第 **n1**,**n2**,**n3**…. 个元素组成的表
t[[**i**,**j**]]	**t** 的第 **i** 个子表的第 **j** 个元素

表的运算函数

函数	意义
Insert[**list**,**expr**,**n**]	给出在表 **list** 的第 **n** 位置插入 **expr** 后的结果
Prepend[**list**,**expr**]	给出在表 **list** 的首部插入元素 **expr** 后的结果
Append[**list**,**expr**]	给出在表 **list** 的末尾插入元素 **expr** 的表
AppendTo[**list**,**expr**]	在表 **list** 的末尾插入元素 **expr**
Take[**list**,**n**]	取 **list** 中的前 **n** 个元素
Take[**list**,{**m**,**n**}]	取从 **m** 到 **n** 的元素(包括 **m**,**n** 在内)
Rest[**list**]	给出删掉第一个元素之后的集合
Drop[**list**,{**m**,**n**}]	给出删掉 **m** 到 **n** 个元素之后的集合
ReplacePart[**list**,**expr**,**n**]	用 **expr** 替换 **list** 中 **n** 处的元素
Length[**list**]	给出表中元素的数目
Sort[**list**]	将表 **list** 的元素按照标准顺序排序
Union[**list**]	去掉重复的元素后对元素排序
Reverse[**list**]	将表 **list** 中元素的顺序倒过来
Apply[**Plus**,**list**]	把 **list** 中的所有元素加在一起
Apply[**Times**,**list**]	把 **list** 中的所有元素乘在一起

例如：

```
In[1]:=a={5,8,5,7,2,6};
In[2]:=Apply[Plus,a](*将 a 中所有元素相加*)
Out[2]=33
In[3]:=Sort[a](*将表 b 的元素由小至大顺序排列*)
Out[3]={2,5,5,6,7,8}
In[4]:=Reverse[%](*反转上表,即表元素由大至小排列*)
Out[4]={8,7,6,5,5,2}
In[5]:=AppendTo[a,9](*定义后的表仍可扩充*)
Out[5]={5,8,5,7,2,6,9}
```

表的集合运算

要进行数学上的集合运算，可用 **Union[list1,list2,…]** 计算 list1,list2,…的和集，用 **Intersection[list1,list2,…]** 计算 list1,list2,…的交集，用 **Complement[un,list1,list2,…]** 计算 un 对 list1,list2,…的补集.

函数	意义
Union[list1,list2,…]	给出表 list1,list2,…中不同元素的一个表
Intersection[list1,list2,…]	给出表 list1,list2,…中公共元素的一个表
Complement[un,list1,list2,…]	给出在 un 中但不在表 list1,list2,…中元素的一个表

● 初等函数

实变量的数值函数	意义
Round[x]	最接近 x 的整数
Floor[x]	不大于 x 的最大整数
Ceiling[x]	不小于 x 的最大整数
Abs[x]	x 的绝对值
Sign[x]	符号函数. x>0 时值为 1,x=0 时值为 0, x<0 时值为 −1
Max[x1,x2,…] 或 **Max[s]**	取 x1,x2,…中的最大值. s 为一集合或数组
Min[x1,x2,…] 或 **Min[s]**	取 x1,x2,…中的最小值. s 为一集合或数组

复变量的数值函数	意义(z=x+Iy)
Re[z]	取复数的实部
Im[z]	取复数的虚部
Conjugate[z]	共扼函数
Abs[z]	复数的模
Arg[z]	复数的幅角

变量可为实数或复数的函数	意义
Exp[z]	指数函数，e^z.
Log[z]	以 e 为底的对数函数，ln z
Log[b,z]	以 b 为底的对数函数，$\log_b z$
Sin[z],Cos[z],Tan[z],Cot[z]	正弦、余弦、正切和余切函数
Csc[z],Sec[z]	余割，正割
ArcSin[z],ArcCos[z],ArcTan[z], ArcCot[z],ArcCsc[z],ArcSec[z]	反三角函数
Sinh[z],Cosh[z],Tanh[z], Coth[z],Csch[z],Sech[z]	双曲函数
ArcSinh[z],ArcCosh[z],ArcTanh[z], ArcCoth[z],ArcCsch[z],ArcSech[z]	反双曲函数

伪随机函数	意义
Random[]	0 到 1 之间的一个伪随机实数
Random[Real,xmax,n]	0 到 xmax 之间的一个 n 位精度的伪随机数
Random[Real, {xmin,xmax}]	xmin 到 xmax 之间的一个伪随机数
Random[Complex]	单位正方形中的一个伪随机复数
Random[Complex, {zmin,zmax}]	由 zmin 和 zmax 确定的长方形中的一个伪随机复数
Random[Integer]	0 或 1 的随机数
Random[Integer, {imin,imax}]	imin 到 imax 之间的一个伪随机整数

整数和组合函数	意义
Mod[m,n]	**m** 被 **n** 除的正余数
Quotient[m,n]	**m/n** 的整数部分
GCD[n1,n2,…]或 **GCD[s]**	**n1,n2,…**的最大公因子,**s** 为一数据集合
LCM[n1,n2,…]或 **LCN[s]**	**n1,n2,…**的最小公倍数,**s** 为一数据集合
Prime[k]	第 **k** 个素数
PrimeQ[n]	当 **n** 为是素数时为 **True**,否则为 **False**
n!	阶乘 $n(n-1)(n-2)\cdots 1$
n!!	双阶乘 $n(n-2)(n-4)\cdots$
Binomial[n,m]	二项式系数 C_n^m
Permutations[list]	列出 **list** 中元素的所有排列
Multinomial[n1,n2,…]	多项式系数
BernoulliB[n]	伯努利系数 B_n
BernoulliB[n]	伯努利多项式 $B_n(x)$
Euler[n]	欧拉数 E_n
Euler[n,x]	欧拉多项式 $E_n(x)$

例:

In[1]:=**Binomial[7,2]**(* 计算 C_7^2 *)

Out[1]=21

In[2]:=**Permutations[{1,2,5}]**(* 列出{**1,2,5**}的所有排列 *)

Out[2]={{1,2,5},{1,5,2},{2,1,5},{2,5,1},{5,1,2},{5,2,1}}

● 算术运算符和表达式

算术表达式由常数、变量、函数、算术运算符和括号组成.常数和变量的类型有:整型、有理数、实型、复数型、列表、向量和矩阵.函数包括系统定义的函数、用户定义的函数和程序包中的函数.其中:方括号[]内放函数变量,花括号{}表示元素的分界符,用圆括号()组织运算量之间的顺序.Mathematica 中的算术运算符+,-,*,/,^ 分别表示加号、减号(或负号)、乘号、除号和乘方;%(百分号)表示上一个计算结果;%%表示上上个计算结果.

在表达式中,乘号可用空格符号代替.在两个变量或数值之间放一个空格即表示求这两个量的乘积.在不引起误解的情况下,乘号可以省略.例如:**2a**,**2*a**,**2 a**的意义是相同的.**(a-b)(c+d)**与**(a-b)*(c+d)**意义也是相同的.算术运算的顺序遵守数学中的习惯,次序为^ → *,/ → +,-.同级运算符遵守从左到右的顺序.

● 自定义函数

f[x]=表达式　　对指定的对象 f[x]定义,x 表示某一具体对象

f[x_]:=表达式　　对函数 f[x]定义,x_表示变量

f[x_]=.　　取消 f[x_]的定义

Clear[f]　　清除 f 的所有定义

出现在 f[x_]中的 x_是一类重要实体,在 Mathematica 中读作"空白",称为模式,它表示函数定义中的变量. 模式 f[x_]可以代表任何形如 f[anything]形式参数的表达式. 如果在定义中只用 f[x]=表达式,那么这个规则仅对对象 x 才起作用.

例如:

```
In[1]:=f[x_]:=2x-1(* 或 f[x_]=2x-1 *)
In[2]:={f[10],f[0],f[y]}
Out[2]={19,-1,-1+2y}
In[3]:=g[x]=2x-1
Out[3]=2x-1
In[4]:=g[10]+g[x](* f[x]与 g[x]的区别在哪里? *)
Out[4]=-1+2x+g[10]
```

如果在定义函数时需要几个命令才能完成工作,可将几个命令依次排列,命令之间用逗号分隔,用圆括号把首尾命令括起来. 如果在定义函数时还要用局部变量,可用 Module 模块.

```
In[5]:=a={1,2,3,7,9};
In[2]:=h[x_]:=(y=Max[x];z=Min[x];y^2+z^2)
In[3]:=h[a]
Out[3]=82
```

二、在 Mathematica 中作图

● 二维函数作图

Plot[函数 f,{x,xmin,xmax},选项]

在区间{x,xmin,xmax}上,按选项的要求画出函数 f 的图形.

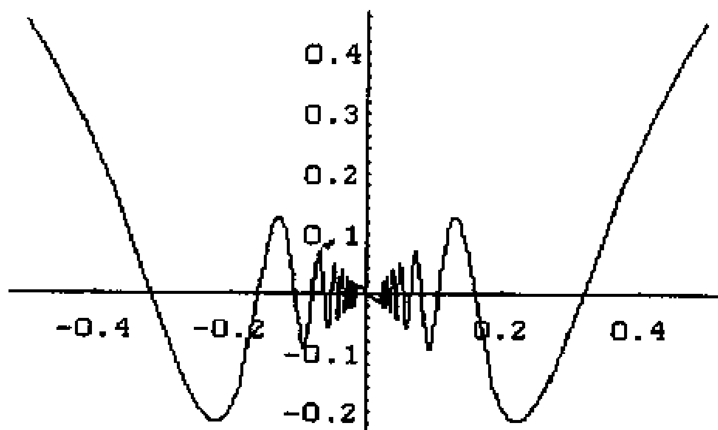
Plot[{函数 1,函数 2},{x,xmin,xmax},选项]

在区间{x,xmin,xmax}上,按选项的要求同时画出几个函数的图形.

Mathematica 绘图时允许使用选项对绘制图形的细节提出各种要求和设置. 例如:要求取消坐标轴,给图形加框线等要求. 每个选项都有一个确定的名

字，以“选项名->选项值”的形式放在 **Plot** 中最后边位置，如 In[3]所示. 一次可设置多个选项，选项依次排列，以逗号相隔.

In[1]:=**Plot[xSin[1/x],{x,-1/2,1/2}]**



（* 图 16-4 *）

Out[1]=-Graphics-

In[2]:=**Plot[(x^2-x)Sin[x],{x,2,16},AxesLabel->{"x","f(x)"}]**

（* 给 x,y 坐标轴分别加标记"x","f(x)",请上机观看 *）

In[3]:=**Plot[Sin[x],{x,0,3},Frame->True,GridLines->Automatic]**

（* 给图形加上框线和网格，请上机观看 *）

下面列出部分选项以及它的意义：

AspectRatio 图形的高度与宽度的比例，默认值是 **1/GoldRatio=0.618** 如果要图形按实际情况显示，设置的选项值是 **Automatic.**

Axes 是否画坐标轴以及设置坐标轴的中心位置，默认值是 **True**，画出坐标轴. **Axes->None** 不设坐标轴；**Axes->{x0,y0}** 设置坐标轴中心为{**x0,y0**}.

AxesLabel 设置坐标轴上的标记符号. 默认值是 **None**，不做标记. 用{“字符串 1”,“字符串 2”}的形式定义轴的横坐标和纵坐标标记.

Frame 在图形周围是否加框. 默认值是 **False**；**Frame->True** 画出边框.

Ticks 设置坐标轴上刻度的位置，默认值是 **Automatic**，由系统自动定位. **Ticks->None** 不标坐标刻度；**Ticks->{xi,yi}** 规定 x 轴和 y 轴的刻度值，**Ticks->{t1,t2,…}** 表示要在一个轴上按 **t1,t2,…**，设置坐标轴刻度.

FrameLabel 是否在框的周围加标志. 默认值是 **None.**

PlotLabel 图形的名称标志. 默认值是 None,不列标志.

用?? **Plot** 可以看到 **Plot** 所有的选项以及选项的值.

我们称曲线的颜色、曲线的线形和线的宽度等特性为曲线样式. 下列选项用于设置曲线的样式.

GrayLevel[g] 灰度比值,g 取 0 到 1 之间的数. 0 为白色,1 为黑色.

RGBColor[r,g,b] 红、绿、兰三色的强度,r,g 和 b 取 0 到 1 之间的数.

Thickness[t] 显示线的宽度为 t,以占整个图的宽度的比来量度.

Dashing[{d1,d2,…}] 用虚线段序列画线.

PointSize[d] 给出一个点的大小 d.

PlotStyle->s1 为所有曲线规定一种线型.

PlotStyle->{{s1},{s2},…} 为一曲线序列循环地使用线样式.

请上机观看 In[1]和 In[2]的运行结果:

In[1]:=**Plot[{x,x^2},{x,-10,10},PlotStyle->{{GrayLevel[0.5]},{RGBColor[0,1,1]}}]**

In[2]:=**Plot[{x,2x},{x,1,3},PlotStyle->{{Thickness[0.01]},{Tbickness[0.05]}}]**

In[3]:=**Plot[{Sin[2x],x},{x,-1.5,1.5},AspectRatio->Automatic,PlotStyle->Dashing[{0.02,0.02,0.01,0.02}]]**

重画和组合图形

Show[pic] 显示图形 pic

Show[pic,选项名->选项值] 设置图形 pic 的各种选项并显示图形

Show[pic1,pic2,…,picn] 将图 pic1,pic2,…,picn 在一起显示

通过使用 **Show** 和一系列不同的选项,可以用多种不同方式察看同一个图形,再找出选项的最佳设置值,并可组合显示多个图形. **Show** 可再现各类作图命令完成的图形.

● 两维参数画图函数

ParametricPlot[{x[t],y[t]},{t,t0,t1},选项] 画一个 x 轴,y 轴坐标为{x[t],y[t]},参变量 t 在[t0,t1]中的参数曲线

ParametricPlot[{{x1[t],y1[t]},{x2[t],y2[t]}},{t,下限,上限},选项] 画一组参数曲线

在 **Plot** 中使用的选项都可用在 **ParametricPlot** 中. 请上机观察下例.

In[1]:=**ParametricPlot[{Sin[t],Sin[2t]},{t,0,2Pi}]**

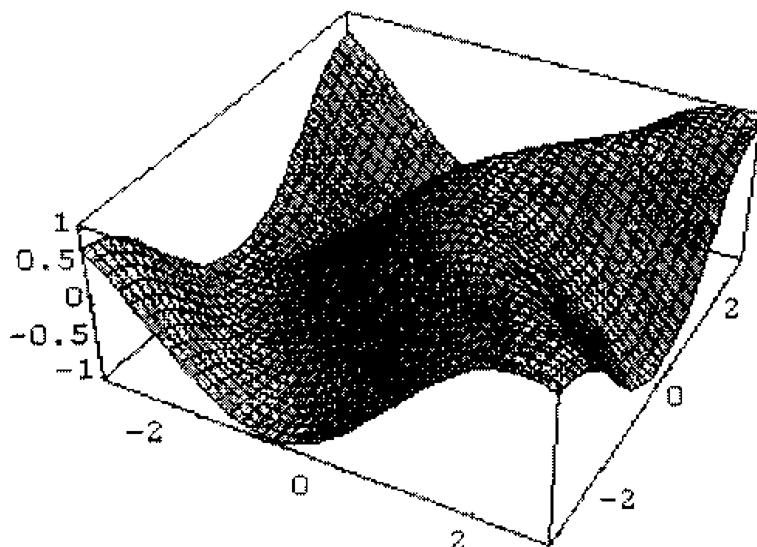
In[2]:=**ParametricPlot[{Cos[t],Sin[t]},{t,0,2Pi},AspectRatio->Automatic]]**

● 三维函数画图

Plot3D[f[x,y],{x,x0,x1},{y,y0,y1},选项]

在区域 x∈[x0,x1]和 y∈[y0,y1]上，画出空间曲面 f[x,y]. f[x,y]为实数值表达式.

In[1]:=**Plot3D[Sin[x Cos[y]],{x,−3,3},{y,−3,3},PlotPoints −> 40]**



(*图 16−5*)

当 **Plot3D** 画一个三维图形时，它将这个目标放在一个透明的长方体盒子中. 取用默认值 **Boxed−>True**，显示这个盒子的边框. 设置选项 **BoxRatios** 能使盒子在不同的方向压缩或拉长. **ViewPoint** 是一个重要的选项，相当于拍摄图形的照相机放在什么位置. 不同的位置看到曲面的形式效果大不一样.

ViewPoint 的典型设置如下：

{0,−2,0}	正前方	{0,0,2}	正上方
{0,−2,2}	前上方	{0,−2,−2}	前下方
{−2,−2,0}	盒左角	{2,−2,0}	盒右角

● 三维参数画图

ParametricPlot3D[{x,y,z},{u,u0,u1,(du)},{v,v0,v1,(dv)},选项]

画三维参数空间曲面,x=x(u,v),y=y(u,v),z=z(u,v),u∈[u0,u1],v∈[v0,v1]

ParametricPlot3D[{x,y,z,s},{u,u0,u1,(du)},{v,v0,v1,(dv)},选项]

按照函数关系画出三维参数空间曲面的阴影部分,x=x(u,v),y=y(u,v),z=z(u,v),s=s(u,v),u([u0,u1],v∈[v0,v1]

ParametricPlot3D[{x,y,z},{t,t0,t1},选项]

画三维参数空间曲线,x=x(t),y=y(t),z=z(t),t∈[t0,t1]

In[1]:=**ParametricPlot3D**[{**Cos**[u](3+**Cos**[v]),**Sin**[u](3+**Cos**[v]),**Sin**[v]},{u,0,2Pi},{v,0,2Pi}]

In[2]:=**ParametricPlot3D**[{u,v,u^3-3uv^2},{u,-1.5,1.5},{v,-1.5,1.5},**Boxed**->**False**,**AspectRatio**->1,**Axes**->**None**,**ViewPoint**->{1,3,1}](* In[1]的图略 *)

● 等值线图和密度图

ContourPlot[f[x,y],{x,xmin,xmax},{y,ymin,ymax},选项]

作二元函数 f[x,y]在区域上的等值线图,等值线很象地图上的等高线,它们把曲面上高度相等的各点连接起来,等值线系列对应于均匀间隔的 z=f(x,y)值数列. 函数 **ContourPlot** 用于作二元函数的等值线图.

执行 **ContourPlot** 以后,Mathematica 送回一个 **ContourGraphics** 目标. 如果函数值的网络不够细,等值线图可能会有误差,当函数值变化幅度较大时,**ContourPlot** 能给出规则的等值线图,当函数值变化太小曲面几乎是平面时,可能给出不规则的等值线图.

Densityplot[f[x,y],{x,xmin,xmax},{y,ymin,ymax},可选项]

作二元函数 f[x,y]在区域上的密度图,密度图与等值线图的作用相似. 在密度图中,相等的数值用同一灰度表示. 执行 **DensityPlot** 以后,Mathematica 送回一个 **DensityGraphics** 目标.

In[1]:=**ContourPlot**[x^2/4+9y^2-6,{x,-2,2},{y,-3,3}](* 请上机观看 *)

In[2]:=**DensityPlot**[**Cos**[x]**Cos**[y],{x,-2,2},{y,-2,2}](* 请上机观看 *)

● 数据绘图

Mathematica 也有直接调用数据画图形的命令，还可以使用 Fortran 或 C 等其它语言生成的数据做图.

画图函数	意义
ListPlot[{{x1,y1},{x2,y2},…}]	画出数据点{**x1,y1**},{**x2,y2**},…
ListPlot[{y1,y2,…,yn}]	画出数据点{{**1,y1**},{**2,y2**}, …,{**n,yn**}}
ListPlot[数据,PlotJoined->True]	画一条通过数据点的光滑曲线
{{**x1,y1,z1**},{**x2,y2,z2**},…}	三维数据数据点{**xi,yi,zi**}, **i=1,2,…,n**
ListPlot3D[数据]	使用数据画出三维图
ListContourPlot[数据]	使用数据画出等值线图
ListDensityPlot[数据]	使用数据画出密度图

请上机观看 In[1],In[2],In[3]的输出.

In[1]:=**t=Table[i^(2/3),{i,1,5,0.2}];ListPlot[t]**

In[2]:=**ListPlot[t,PlotJoined->True]**

In[3]:=**Table[{Random[],Random[]},{i,12}];ListPlot3D[%]**

在 **graphics** 程序包中调入 **graphics.m**,可作一些常规的棒图和饼图等图形.

In[1]:=**<<"packages\graphics\graphics.m"**

In[2]:=**d=Table[Prime[n],{n,10}];BarChart[d]**(*请上机观看*)

In[3]:=**PieChart[d]**

● 用图形元素画图

使用图形元素做图适合于画结构复杂的图形.在 Mathematica 中也提供了各种二维和三维图形元素函数,如点、圆弧和立方体等.先用 **Graphics**[图形元素]做出平面图形表达式,再用 **Show**[图形表达式]的形式显示完成的图形.前面在 **Show** 中用过的选项现在仍然能用.

二维图形元素	几何意义
Point[{**x**,**y**}]	点的位置在{**x**,**y**},**x** 和 **y** 为坐标值
Line[{{**x1**,**y1**},{**x2**,**y2**},…}]	依次连接相邻两点的线段
Rectangle[{**xmin**,**ymin**}, {**xmax**,**ymax**}]	以{**xmin**,**ymin**}和{**xmax**,**ymax**}为对角线坐标的填实矩形
Polygon[{**x1**,**y1**},{**x2**,**y2**},…]	以{**x1**,**y1**},{**x2**,**y2**},…为顶点的封闭多边形
Raster[{{**a11**,**a12**}, {**a21**,**a22**},…}]	灰度颜色的矩阵
Circle[{**x**,**y**},**r**]	圆心在{**x**,**y**},半径为 **r** 的圆
Circle[{**x**,**y**},{**rx**,**ry**}]]	圆心在{**x**,**y**},长短半轴为 **rx** 和 **ry** 的椭圆
Circle[{**x**,**y**},**r**,{**t1**,**t2**}]	从弧度 **t1** 到弧度 **t2** 的圆弧
Circle[{**x**,**y**},{**rx**,**rt**},{**t1**,**t2**}]	从弧度 **t1** 到弧度 **t2** 的椭圆弧
Disk[{**x**,**y**},**r**]	圆心在{**x**,**y**},半径为 **r** 的填实圆

例如:

In[1]:=**Graphics**[{**Line**[{{−1.5,−1.5},{1.5,1.5}}],**PointSize**[**0.03**],**Point**[{**0**,**1**}],**Point**[{**1**,**0**}]}]

Out[1]=−Graphics−(*并不显示图形*)

In[2]:=**Show**[%](***Show** 又发挥作用了,请上机观看*)

三维图形元素	几何意义
Point[{**x**,**y**,**z**}]	点{**x**,**y**,**z**}
Line[{{**x1**,**y1**,**z1**}, {**x2**,**y2**,**z2**},…}]	通过点{**x1**,**y1**,**z1**},{**x2**,**y2**,**z2**},…的线
Polygon[{{**x1**,**y1**,**z1**}, {**x2**,**y2**,**z2**},…}]	具有指定角的填实多边形
Cuboid[{**x0**,**y0**,**z0**},{**x1**,**y1**,**z1**}]	以{**x0**,**y0**,**z0**}和{**x1**,**y1**,**z1**}为对角线的立方体
Text[**expr**,{**x**,**y**,**z**}]	在{**x**,**y**,**z**}处的文本

例如:

In[1]:=**p**=**Table**[**Point**[{**Random**[],**Random**[],**Random**[]}],{**24**}];

In[2]:=**Show**[**Graphics3D**[**p**]]

三、初等代数运算

● 多项式运算

多项式的展开

多项式是一类特殊形式的表达式.表达式中的各种运算都可用于多项式的运算中.表达式中的各种输出形式也都可用于多项式的输出.**Expand** 将多项式按幂次由低至高展开成单项之和;**FactorTerms** 提取每个元素的公因子;**Factor** 做因式分解,将多项式写成尽可能小的因式之积;**Simplify** 化简多项式使其包含的项数最少意义下的最简形式.

函数	意义
Expand[expr]	按幂次展开多项式 **expr**
Factor[expr]	对多项式 **expr** 因式分解
FactorTerms[expr]	提出多项式 **expr** 各项中的公因子
Collect[expr,x]	把 **expr** 写成 **x** 的幂次之和
Collect[expr,{x,y,…}]	把多项式写成 **x,y,…** 的幂次之和
Simplify[多项式]	把多项式写成最简形式

In[1]:=**p=Expand[(x+2y+1)^2]**(* 展开 $(x+2y+1)^2$ *)

Out[1]=1+2x+x²+4y+4xy+4y²

In[2]:=**Factor[x^6-y^6]**(* 对 x^6-y^6 因式分解 *)

Out[2]=(x-y)(x+y)(x²-xy+y²)(x²+xy+y²)

In[3]:=**Collext[p,y]**(* 按 y 的幂次排列 $(x+2y+1)^2$ *)

Out[3]=1+2x+x²+(4+4x)y+4y²

In[4]:=**Length[p]**(* 计算多项式 p 的长度 *)

Out[4]=6

In[5]:=**p[[2]]**(* 表达式 p 的第 2 项 *)

Out[5]=2x

有理多项式展开函数	意义
ExpandNumerator[expr]	只展开有理式的分子
ExpandDenominator[expr]	只展开有理式的分母
Expand[expr]	展开分子,每项除以分母
Factor[expr]	完全分解
Together[expr]	把所有项合成在一个公共分母上的分式
Apart[expr]	将表达式分解成部分分式之和
Cancel[expr]	约去分子分母的公因式

多项式的代数运算多项式的代数运算有加法(+),减法(—),乘法(*)和除法(/).其中多项式的加法、减法和乘法的运算符号与代数中运算符号的相同,除号略有不同.如果用除号"/"做多项式除法,最多约去明显分子分母的公因子.在做多项式除法时要用函数 **PolynomialQuotient** 或 **PolynomialRemainder** 才能得到得到商式或余式

多项式运算函数	意义
PolynomialQuotient[p,q,x]	计算关于 x 的多项式 p 和 g 相除的商式
PolynomialRemainder[p,q,x]	计算多项式 p 和 g 相除的余式
PolynomialQuotientRemainder[q,p,x]	计算多项式 p 和 g 相除的商式和余式
PolynomialGCD[多项式 1,多项式 2,…]	计算多项式 1,多项式 2,…的最大公因子
PolynomialLCM[多项式 1,多项式 2,…]	计算多项式 1,多项式 2,…的最小公倍数
Resultant[多项式 1,多项式 2,x]	计数两多项式的结式
Cyclotomic[n,x]	给出关于 x 的 n 阶割圆多项式 $C_n(x)$

● 方程求根

解方程函数 **Solve** 的一般形式:

Solve[方程或方程组,{变量表列}]

NSolve[方程或方程组,{变量表列}]

Solve 的目标是找出方程的精确解,Mathematica 总可以解出四阶或四阶以下多项式方程的精确解.对于三次或四次方程,用 **Solve** 算出的结果可能相当复杂.在解四次以上的多项式方程时,Mathematica 碰到数学理论上的障碍而不能给出精确解的表达式;函数 **NSolve** 是找出方程的数值解,因而总能解出方程或方程组的数值解.

数学中方程式表示为:$x^2-6x-5=0$.在 Mathematica 中等号(=)作为赋值号或定义变量的值,代数方程中的等号使用逻辑等号"=="表示,如:**x^2-6x-5==0**.这样,Mathematica 中的方程表示成一个逻辑表达式,它的意义是方程中的等号两边的量是否相等.

In[1]:=**Solve[x^5+9x+1==0,x]**(* 找不到方程的精确解 *)

Out[1]={ToRules[Roots[9x+x^5==-1,x]]}

In[2]:=N[%](*或用 NSolve[x^5+9x+1==0,x],直接求数值解*)

Out[2]={{x->-1.19618-1.2256I},{x->-1.19618+1.2256I},{x->-0.111109},{x->1.25173-1.22546I},{x->1.25173+1.22546I}}

In[3]:=Solve[{3x-2y==5a,x+y==5b},{x,y}] (* 解方程 $\begin{cases}3x-2y=5a\\x+y=5b\end{cases}$ *)

Out[3]={{x->a+2b,y->-a+3b}}

● 和与积

和与积的函数	数学意义
Sum[f_i,{i,min,max}]	计算和式:$\sum_{i=\min}^{\max} f_i$
Sum[f_i,{i,min,max,di}]	以步长 di 计算和式
Sum[$f_{i,j}$,{i,i0,i1},{j,j0,j1}]	计算两重和式:$\sum_{i=i0}^{i1}\sum_{j=j0}^{j1} f_{i,j}$
Product[f_i,{i,min,max}]	计算乘积式:$\prod_{i=\min}^{\max} f_i$
Product[f_i,{i,min,max,di}]	以步长 di 计算 f_i 的乘积
Product[$f_{i,j}$,{i,i0,i1,s},{j,j0,j1,js}] 注:is 和 js 为步长,其值是 1 时可省略	计算两重乘积式:$\prod_{i=i0}^{i1}\prod_{j=j0}^{j1} f_{i,j}$

还有函数 NSum[fn,循环范围]和 NProduct[fn,循环范围]. Sum 与 NSum 的区别正如 Solve 与 NSolve 的区别一样. Sum 用于计算和式的精确值, NSum 计算和式的近似数值;同理,可知 Product 与 NProduct 的区别.

In[1]:=Sum[x^n/n!,{n,0,5}] (* 计算 $\sum_{n=0}^{5}\frac{x^n}{n!}$ *)

Out[1]=$1+x+\frac{x^2}{2}+\frac{x^3}{6}+\frac{x^4}{24}+\frac{x^5}{120}$

In[2]:=Sum[1/k^3,{k,1,10}] (* 计算 $\sum_{k=1}^{10}\frac{1}{k^3}$ *)

Out[2]=$\frac{19164113947}{16003008000}$ （*Sum 给出和的精确值*）

In[3]:=**NSum[1/k^3,{k,1,Infinity}]** $\left(* 计算\sum_{k=1}^{\infty}\frac{1}{k^3}*\right)$

Out[3]=1.20206

In[4]:=**NProduct[Sin[k],{k,1,5,2}]**

Out[4]= 0.113871

In[5]:=**Product[(u-6)(v+6+s),{u,1,3},{v,1,3}]**

Out[5]=-21600(7+s)^3(8+s)^3(9+s)^3

四、微积分

● 求极限

计算函数极限 $\lim_{x \to x_0} f(x)$ 的一般形式是：

Limit[expr,x->x0] $x \to x_0$ 时函数 **expr** 的极限

Limit[expr,x->x0,Direction->1] $x \to x_0^-$ 时函数 **expr** 的极限

Limit[expr,x->x0,Direction->-1] $x \to x_0^+$ 时函数 **expr** 的极限

In[1]:=**Limit[(x^2-1)/(4x^2-7x+1),x->Infinity]**(* 计算 $\lim_{x\to\infty}\frac{x^2-1}{4x^2-7x+1}$ *)

Out[1]=$\frac{1}{4}$

In[2]:=**Limit[Sin[1/x],x->0]**

Out[2]=RealInterval[{-1,1}](*在(-1,1)之间振荡*)

In[3]:=**Limit[1/x,x->0,Direction->-1]**(* 计算 $\lim_{x\to 0^+}\frac{1}{x}$ *)

Out[3]=Infinity

● 微商和微分

在 Mathematica 中能方便地计算任何函数表达式的任意阶微商(导数)。如果 f 是一元函数，D[f,x]表示 $\frac{df(x)}{dx}$；如果 f 是多元函数，D[f,x]表示 $\frac{\partial}{\partial x}f$。微商函数的常用形式如下：

D[f,x] 计算偏导数 $\frac{\partial}{\partial x}f$

D[f,x1,x2,…] 计算多重导数 $\frac{\partial}{\partial x1}\frac{\partial}{\partial x2}\cdots f$

D[f,{x,n}] 计算 n 阶导数 $\frac{\partial^n}{\partial x^n}f$

D[f,x,NonConstants->{v1,v2,…}] 计算 $\frac{\partial}{\partial x}f$，其中 **v1,v2,…**依赖于 **x**

In[1]:=**D[x^ x,x]**(* 计算 x^x 的一阶导数 *)

Out[1]=x^x+x^xLog[x]

In[2]:=**D[z Sin[x^ 2 y^ 2],x,y]**(* 计算$\frac{\partial}{\partial x\partial y}z\sin(x^2y^2)$ *)

Out[2]=4x y zCos[x²y²]-4x³ y³Sin[x² y²]

下面列出全微分函数 Dt 的常用形式及其意义：

Dt[f] 全微分 $\mathrm{d}f$

Dt[f,x] 全导数$\frac{\mathrm{d}f}{\mathrm{d}x}$

Dt[f,x1,x2,…] 多重全导数$\frac{\mathrm{d}f}{\mathrm{d}x1}\frac{\mathrm{d}f}{\mathrm{d}x2}\cdots f$

Dt [f, x, Constants -> {c1, c2, …}] 全导数，说明 ci 为常数 $\left(\text{即}\frac{\mathrm{d}(ci)}{\mathrm{d}x}=0\right)$

y/ :Dt[y,x]=0 置$\frac{\mathrm{d}y}{\mathrm{d}x}=0$

In[3]:=**Dt[x^ 2+y^ 2]**

Out[3]=2x Dt[x]+2y Dt[y]

In[4]:=**Dt[x^ 2+y^ 2,x]** $\left(\text{* 计算全导数}\frac{\mathrm{d}}{\mathrm{d}x}(x^2+y^2)\text{ *}\right)$

Out[4]=2x+2y Dt[y,x]

● 不定积分和定积分

不定积分

Integreate 函数主要计算只含有“简单函数”的被积函数.“简单函数”包括有理函数、指数函数、对数函数和三角与反三角函数.不定积分一般形式如下：

Integrate[f,x] 计算不定积分$\int f(x)\mathrm{d}x$

Integrate[f,x,y] 计算不定积分$\int \mathrm{d}x\int f(x,y)\mathrm{d}y$

Integrate[f,x,y,z] 计算不定积分$\int \mathrm{d}x\int \mathrm{d}y\int f(x,y,z)\mathrm{d}z$

In[1]:=**Integrate[1/(x^ 2-1),x]**(* 计算$\int\frac{1}{x^2-1}\mathrm{d}x$ *)

Out[1]=$\frac{\text{Log}[1-x]}{2}-\frac{\text{Log}[1+x]}{2}$

In[2]:=**Integrate[3x^2+y,x,y]**(* 计算$\iint(3x^2+y)\mathrm{d}x\mathrm{d}y$ *)

Out[2]=$3x^3+2y^2$

Mathematica 除了能计算简单积分外，还能计算一类如贝塞尔(**BesselJ[n,z]**)、伽马(**Gamma[z]**)等二三十个数学物理特殊函数.这些特殊函数在很多应用范围内扩充了 Mathematica 的积分功能.

例如:**Gamma[z]**函数定义的积分 $\Gamma(z)=\int_0^{\infty}t^{a-1}\mathrm{e}^{-t}\mathrm{d}t$

Beta[a,b]函数定义的积分 $B(a,b)=\frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}=\int_0^1 t^{a+1}(1-t)^{b-1}\mathrm{d}t$

还可以通过定义函数或程序包来增强 Mathematica 积分的功能.

定积分

计算定积分的命令和计算不定积分是同一个 **Integrate** 函数，在计算定积分时，除了要给出变量外还要给出积分的上下限.当定积分算不出准确结果时，用 N[%]命令总能得到其数值解. **NIntegrate** 也是计算定积分的函数，其使用方法和形式和 **Integrate** 函数相同.用 **Integrate** 函数计算定积分得到的是准确解，**NIntegrate** 函数计算定积分得到的是近似数值解.计算多重积分时，第一个自变量相应于最外层积分放在最后计算.

Integrate[f,{x,a,b}]　计算定积分$\int_a^b f(x)\mathrm{d}x$

NIntegrate[f,{x,a,b}]　计算定积分$\int_a^b f(x)\mathrm{d}x$

Integrate[f,{x,a,b},{y,c,d}]　计算定积分$\int_a^b \mathrm{d}x\int_c^d f(x,y)\mathrm{d}y$

NIntegrate[f,{x,a,b},{y,c,d}]　计算定积分$\int_a^b \mathrm{d}x\int_c^d f(x,y)\mathrm{d}y$

In[1]:=**Integrate[Cos[x]^2+Sin[x]^3,{x,0,1}]** (* 计算$\int_0^1(\cos^2 x+\sin^2 x)\mathrm{d}x$ *)

Out[1]=$\frac{2}{3}+\frac{6-9\mathrm{Cos}[1]+\mathrm{Cos}[3]+3\mathrm{Sin}[2]}{12}$

In[2]:=**NIntegrate[Cos[x]^2+Sin[x]^3,{x,0,1}]**

Out[2]=0.906265(* 用 **Nintegrate** 计算定积分的数值解 *)

In [3]:=**Integrate[x+y,{x,b,a},{y,0,x}]**(* 计算$\int_b^a\int_0^x(x+y)\mathrm{d}y\mathrm{d}x$ *)

Out[3]=$\frac{a^3}{2}-\frac{b^3}{2}$

● 幂级数

幂级数展开函数 Series 的一般形式：

Series[expr, {x, x0, n}] 将 expr 在 x=x0 点展开到 n 阶的幂级数

Series[expr, {x, x0, n}, {y, y0, m}] 先对 y 展开到 m 阶再对 x 展开 n 阶幂级数

用 Series 展开后，展开项中含有截断误差相 O[x]ⁿ，请看下面实例：

In[1] :=**Series[Sin[2x], {x, 0, 6}]**

Out[1]= $2x-\frac{4x^3}{3}+\frac{4x^5}{15}+O[x]^7$

In[2] :=**Series[f[x], {x, 0, 3}]**

Out[2]= $f[0]+f'[0]x+\frac{f''[0]x^2}{2}+\frac{f^{(3)}[0]x^3}{6}+O[x]^4$

In[3] :=**Series[Exp[x]^ 1/2, {x, -1, 3}]**

Out[3]= $\frac{1}{2E}+\frac{1+x}{2E}+\frac{(1+x)^2}{4E}+\frac{(1+x)^3}{12E}+O[1+x]^4$

In[4] :=**Series[Cos[x]Cos[y], {x, 0, 3}, {y, 0, 3}]**

Out[4]= $1-\frac{y^2}{2}+O[y]^4+(-\frac{1}{2}+\frac{y^2}{4}+O[y]^4)x^2+O[x]^4$

Series 在处理多元函数幂级数时，同 **Integrate** 和 **Sum** 等一样，从最后一个变量到第一个变量逐个展开.

● 常微分方程

求解常微分方程和常微分方程组的函数的一般形式如下：

DSolve[eqns, y[x], x] 解 y(x)的微分方程或方程组 eqns, x 为变量

DSolve[eqns, y, x] 在纯函数的形式下求解

NDSolve[eqns, y[x], {x, xmin, xmax}] 在区间{xmin, xmax}上求解变量是 x 的常微分方程或联立常微分方程组 eqns 的数值解

In[1] :=**DSolve[y'[x]==a y[x], y[x], x]**(* 解微分方程 $y'(x)=ay(x)$ *)

Out[1]={{y[x]->E^{ax}C[1]}}

In[2] :=**DSolve[{y'[x]==a y[x], y[0]==1}, y[x], x]**(* 解微分方程 $y'(x)=ay(x)$，边界条件：$y(0)=1$ *)

Out[2]={{y[x]->E^{ax}}}(* 解常微分方程组：$\begin{cases}x'(t)=y(t)\\y'(t)=x(t)\end{cases}$ *)

In[3] :=**DSolve[{x'[t]==y[t], y'[t]==x[t]}, {x[t], y[t]}, t]**

Out[3]={{x[t]->$\frac{C[1]+E^{2t}C[1]-C[2]+E^{2t}C[2]}{2E^t}$,

y[t]->$\frac{-C[1]+E^{2t}C[1]+C[2]+E^{2t}C[2]}{2E^t}$}}

五、线性代数

● 定义向量和矩阵函数

定义一个矩阵，可用函数 **Table** 或 **Array**. 当矩阵元素能用一个函数表达式表示时，用函数 **Table** 在定义矩阵大小的同时也给每个矩阵元素定义了确定的值. 用函数 **Range** 只能定义元素为数值的向量. **Array** 只能用于定义向量、矩阵和张量，并规定矩阵和张量的元素下标从 **1** 开始. **Array** 的一般形式：

Array[向量元素名,**n**,**f**]

定义下标从 **f** 开始的有 **n** 个元素的向量，当 **f** 是 **1** 时可省略.

Array[矩阵元素名,{**m**,**n**}]

定义 **m** 行 **n** 列的矩阵. 其中：矩阵元素名是一个标识符，表示矩阵元素的名称，当循环范围是{**u**,**v**,**w**}时定义一个张量.

Table[表达式 **f**,循环范围]

表达式 **f** 表示向量或矩阵元素的通项公式；循环范围定义矩阵的大小.

循环范围的一般形式：{循环变量名,循环初值,循环终值,循环步长}.

在 **Array** 或 **Table** 的循环范围表示方法略有区别. 请在下面的实例中注意观察.

In[1]:=**Table**[**a**[**i**,**j**],{**i**,**2**},{**j**,**2**}]

Out[1]={{a[1,1],a[1,2]},{a[2,1],a[2,2]}}(＊矩阵每一行元素用一对{}括起来＊)

In[2]:=**U**=**Array**[**a**,{**2**,**2**}]

Out[2]={{a[1,1],a[1,2]},{a[2,1],a[2,2]}}

In[3]:=**IdentityMatrix**[**3**](＊**IdentityMatrix**[**n**]生成 **n** 维单位矩阵＊)

Out[3]={{1,0,0,},{0,1,1},{0,0,1}}

In[4]:=**DiagonalMatrix**[{**1**,**2**,**3**}](＊生成对角线元素为表元素的对角矩阵＊)

Out[4]={{1,0,0},{0,2,0},{0,0,3}}

In[5]:=**TableForm**[%]

(＊**TableForm**[**m**]或 **MatrixForm**[**m**]按矩阵形式输出 **m**,Out[6]输出略＊)

一个矩阵可用一个变量表示，如 In[2]所示 **U** 是一个矩阵，则 **U**[[**i**]]表示 **U** 的第 **i** 行的 **n** 个元素；**Transpose**[U][[**j**]]表示 **U** 的第 **j** 列的 **m** 个元素；**U**[[**i**,**j**]]或 **a**[**i**,**j**]表示 **U** 的第 **i** 行第 **j** 列元素；**U**[[{**i1**,**i2**,…,**ip**},{**j1**,**j2**,…,

jq}]]表示由行为{i1,i2,…,ip}和列为{j1,j2,…,jq}组成的子矩阵.

● 矩阵的运算符号和函数

表达式	意义
A+c	**A** 为矩阵,**c** 为标量,**c** 与 **A** 中的每个元素相加
A+B	**A**,**B** 为同阶向量或矩阵,**A** 与 **B** 的对应元素相加
cA	**A** 为矩阵,**c** 为标量,**c** 与 **A** 中的每个元素相乘
U.V	向量 **U** 与 **V** 的内积
A.B	矩阵**A** 与矩阵**B** 相乘,要求**A** 的列数等于**B** 的行数
Det[M]	计算矩阵 **M** 行列式的值
Transepose[M]	**M** 的转置矩阵(M^T或 M')
Inverse[M]	计算矩阵 **M** 的逆矩阵(M^{-1})
Eigenvalues[A]	计算矩阵 **A** 的全部(准确解)特征值
Eigenvalues[N[A]]	计算矩阵 **A** 的全部(数值解)特征值
Eigenvectors[A]	计算矩阵 **A** 的全部(准确解)特征向量
Eigenvectors[N[A]]	计算矩阵 **A** 的全部(数值解)特征向量
Eigensystem[A]	计算矩阵**A** 的所有(准确解)的特征值和特征向量
Eigensystem[N[A]]	计算矩阵**A** 的所有(数值解)的特征值和特征向量

● 方程组求解函数

在 Mathematica 中用 **LinearSolve[A,B]**,求解满足方程组 $AX=B$ 的一个解.如果 A 的行列式不为零,那么这个解是方程组的唯一解;如果 A 的行列式是零,那么这个解是方程组的一个特解,方程组的全部解由基础解系向量的线性组合加上这个特解组成.

NullSpace[A]计算方程组 $AX=0$ 的基础解系的向量表,用 **LinearSolve[A,B]**和 **NullSpace[A]**联手解出方程组 $AX=B$ 的全部解.

Mathematica 中还有一个美妙的函数 **RowReduce[A]**,它对 A 的行向量作化简成梯形的初等线性变换.用 **RowReduce** 可计算矩阵的秩,判断向量组是线性相关还是线性无关和计算极大线性无关组等工作.

解方程组函数	意义
RowReduce[A]	作行的线性组合化简 **A**,A 为 m 行 n 列的矩阵
LinearSolve[A,B]	求解满足方程组 $AX=B$ 的一个解,A 为方阵
NullSpace[A]	求解方程组 $AX=0$ 的基础解系的向量表,A 为方阵

例：已知 $A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 1 \\ 3 & 1 & -1 & 3 \\ 3 & 2 & 1 & 3 \end{pmatrix}$ 计算 A 的秩，计算 $AX = \mathbf{0}$ 的基础解系.

In[1]:=**A={{1,1,1,1},{1,0,-1,1},{3,1,-1,3},{3,2,1,3}};**

In[2]:=**RowReduce[A]**

Out[2]={{1,0,-1,1},{0,1,2,0},{0,0,0,0},{0,0,0,0}}(*显然，A的秩是2*)

In[3]:=**NullSpace[A]**

Out[3]={{-1,0,0,1},{1,-2,1,0}}(*A的两个线性无关解*)

六、数值计算方法

● 插值多项式

Interpolation 插值函数的一般形式：

Interpolation[data,InterpolationOrder->n]

对数据 **data** 进行插值运算，并可设置插值多项式的次数 **n**，默认值为 3，**Interpolation** 生成一个 **InterpolatingFunction**[插值范围，<>]目标（如 Out[2]），系统不显示所构造的插值函数. 因此，用户直接用生成的插值函数计算函数的近似值. 例如：

In[1]:=**data=Table[{{1,16},{2,12},{4,8},{5,9}}];**

In[2]:=**g=Interpolation[data,InterpolationOrder->3]**

Out[2]=InterpolatingFunction[{1,5},<>]

In[3]:=**g[1.2]**(*计算 **x=1.2** 处函数的近似值*)

Out[3]=15.1307

在数据中还可以包括插值点处的导数，按{{**x0**,{**y0**,**dy0**}},{**x1**,{**y1**,**dy1**}},…}形式存放数据.

例：已知插值条件，$f(0)=0, f(1)=2, \left.\frac{\mathrm{d}f}{\mathrm{d}x}\right|_{x=0}=1, \left.\frac{\mathrm{d}f}{\mathrm{d}x}\right|_{x=1}=1$，求三次插值多项式.

In[4]:=**d={{0,0,1},{1,2,1}};h=Interpolating[d,x]**

Out[4]=InterpolatingFunction[{0,1},<>]

In[5]:=**h[0.2]**

Out[5]=0.336

● 曲线拟合

拟合函数 **Fit** 的一般形式：

Fit[数据，拟合函数的基，变量]

Fit[data,fun,vars]用数据 **data**，以 **vars** 为变量，按函数 **fun** 的形式构造拟合函数.

例如：设某次实验数据如下：

x	1.36	1.49	1.73	1.81	1.95	2.16
y	14.09	15.09	16.84	17.38	18.44	19.95

试按最小二乘法用一次多项式、二次多项式拟合以上数据.

In[1]:=**f={{1.36,14.09},{1.49,15.09},{1.73,16.84},{1.81,17.38},{1.95,18.44},{2.16,19.95}};Fit[f,{1,x},x]**

Out[1]=4.17841+7.30662 x

In[2]:=**Fit[f,{1,x,x^2},x]**

Out[2]=3.78051+7.77224 x−0.133012 x^2

● 方程求根

Solve 和 **FindRoot** 都能求方程的根，**Solve** 主要用于多项式方程组的求根，**FindRoot** 计算非线性方程或方程组的一个数值解. 下列 **FindRoot** 的求解形式及其意义：

函数 **FindRoot**	意义
FindRoot[方程,{**x,x0**}]	从 **x=x0** 开始，计算方程的一个数值解.
FindRoot[方程,{**x**,{**x0,x1**}}]	以**x0** 和**x1** 为初始值，计算方程的一个数值解. 当找不出**f** 的显式的导数表示时使用.
FindRoot[方程,{**x,xstart, xmin,xmax**}]	在 **xmin,xmax** 范围内计算方程的一个数值解
FindRoot[{方程组},{**x,x0**}, {**y,y0**},…]	计算联立方程组的数值解.

● 数值积分

函数 **Integrate** 和 **NIntegrate**，表面上看起来象一对函数. 前者是计算积分的准确解，即符号计算；后者是计算积分的近似值，即数值计算. 对于被积函数的奇点，可在积分区域上列出.

In[1]:= **NIntegrate[1/Sqrt[Abs[x]],{x,−1,1}]**(* 计算 $\int_{-1}^{1}\frac{1}{\sqrt{|x|}}dx$，0 是奇点 *)

Out[1]=4

例如:计算复平面上的线积分,积分曲线从-1起到-i,1,i再回到1的闭曲线.

In[2]:=**NIntegrate[1/x,{x,-1,-i,1,i,-1}]**

Out[2]=1.11022 10^{-16}+6.28319 I(* 由高斯公式准确解是2πi *)

● 计算极小值的函数

函数	意义
FindMinimum[f,{x,x0}]	以初始点为 **x=x0** 计算 **f** 的一个局部极小值点
FindMinimum[f,{x,{x0,x1}}]	以 **x0,x1** 为初始值计算**f** 极小值,当找不出**f** 的显式导数表示时使用.
FindMinimum[f,{x,{xs,x0,x1}}]	以 **x=xs** 为初始值,在[**x0,x1**]区间计算 **f** 的极小值.
FindMinimum[f,{x,x0},{y,y0},…]	计算多变量函数的极小值,初值为{**x0,y0**,…}

例:In[3]:=**FindMinimum[x^4+3x^2y+x y,{x,0.1},{y,0.2}]**

Out[3]={-0.832579,{x->-0.886325,y->-0.335672}}

在x=-0.886325,y=-0.335672处,f(x,y)的极小值为-0.832579.

● 傅立叶(Fourier)变换

Mathematica可在复数域内进行Fourier变换和反Fourier变换.对于长度为n的表,表的元素a_r的Fourier变换b_s定义为$\frac{1}{\sqrt{n}}\sum_{r=1}^{n}a_r e^{2\pi i(r-1)(s-1)/n}$,注意零频率项在位置1.Fourier变换将数据的时间序列变成数据的频率分量.用Fourier逆变换可以重新得到时间序列.长度为n表的元素为b_s的Fourier逆变换a_r为$\frac{1}{\sqrt{n}}\sum_{s=1}^{n}b_s e^{-2\pi i(r-1)(s-1)/n}$.

不论数据表的长度n是否为2的乘幂,Fourier变换都给出计算结果.

常用的傅立叶变换的形式有:

Fourier[{a0,a1,…,an}]	Fourier 变换
InverseFourier[{b0,b1,…,bn}]	反 Fourier 变换
Fourier[{{a00,a01,…},{a10,a11,…}…}]	二维变换

例如：

In[1]:={-1,-1,-1,-1,1,1,1,1};**ft=Fourier**[%]

(* 对一组方形脉冲做 Fourier 变换 *)

Out[1]={0.,-0.707107-1.70711 I,0.,-0.707107-0.292893 I,
0.,-0.707107+0.292893 I,0.,-0.707107+1.70711 I}

In[2]:=**InverseFourier**[%]

Out[2]={-1.,-1.,-1.+2.7477 10^{-16} I,-1.,1.,1.,1.-2.7477 10^{-16} I,1.}

In[3]:=**Chop**[%]

Out[3]={-1,-1,-1,-1,1,1,1,1}(* 恢复了原数据 *)

● 常微分方程数值解

NDSolve[{eqn1,eqn2,…},y,{x,xmin,xmax}] 对常微分方程 **eqn1**，求函数 **y** 关于 **x** 在[**xmin,xmax**]范围内的数值解.

NDSolve[{eqn1,eqn2,…},{y1,y2,…},{x,xmin,xmax}] 对常微分方程组，求函数 **y1,y2,**…关于 **x** 在[**xmin,xmax**]范围内的数值解.

方程或方程组的初始条件也必须作为方程列出，并和方程放在一起.要解 **n** 阶的常微分方程，必须同时给出 **n-1** 个导数的初始值. **NDSolve** 以 **InterpolatingFunction** 目标生成函数 **y** 的解，并由此得到[**xmin,xmax**]上的任何一点 **x** 的 **y(x)**的近似值.

例:求解常微分方程组 $\begin{cases} x'(t) = -y(t) - x^2(t) \\ y'(t) = 2x(t) - y(t) \\ x(0) = y(0) = 1 \end{cases}, t \in [0,1]$,

计算 $x(1.2)$，$y(1.2)$ 的值.

In[1]:=**NDSolve[{x'[t]==-y[t]-x[t]^ 2,y'[t]==2 x[t]-y[t],**
x[0]==y[0]==1},{x,y},{t,0,3}]

Out[1]={{x->InterpolatingFunction[{0.,3.},<>]},
{y->InterpolatingFunction[{0.,3.},<>]}}

In[2]:={**p=x/.First[%],q=y/.Last[%]**}

Out[2]={InterpolatingFunction[{0.,3.},<>],InterpolatingFunction[{0.,3.},<>]

In[3]:={**p[1.2],q[1.2]**}(* 计算 {$x(1.2)$,$y(1.2)$} *)

Out[3]={-0.301149,0.432898}

● 线性规划

ConstrainedMax[目标函数,{限定区域},{变量}]

ConstrainedMin[目标函数,{限定区域},{变量}] 在限定区域内求线性目标函数的极大或极小值.

LinearProgramming[**c**,**A**,**b**] 计算 $\min f = c'X, s.t.\ AX \geqslant b, X \geqslant 0$,其中,$c' = (c_1, c_2, \cdots, c_n)$,$X - (x_1, x_2, \cdots, x_n)$

例如:计算函数 $f(x,y) = 3x + 2y - 1$,在限定区域:$x < 1, y > 2$ 上的最大值和最小值.

In[1]:=**ConstrainedMax[3x+2y-1,{x<1,y<2},{x,y}]**

Out[1]={6,{x->1,y->2}}(* 在点{1,2}上达最大值 $f(1,2)=6$ *)

In[2]:=**ConstrainedMin[3x+2y-1,{x<1,y<2},{x,y}]**

Out[2]={(1,{x->0,y->0}}(* 在点{0,0}上达最小值 $f(0,0)=-1$ *)

七、过程编程

● 过程的基本构成

简单地说,在 Mathematica 中的一个过程是用分号隔开的表达式序列,一个表达式序列也称为一个复合表达式.在 Mathematica 的定义中,任何一个表达式的位置都能放一个复合表达式.运行时过程中的表达式依次求值.编程中或许更习惯说是语句序列,在一个输入行中就可以放一个过程,调用一个过程就象调用一个函数.

在过程中,经常需要用工作变量保存计算的中间结果,称这类变量为局部变量.局部变量可以在过程中得到任何想要的值,而不会影响过程外部的值.如果要用局部变量与表达式序列组成一个程序块,可用 **Module** 模块结构.

Module[{局部变量表},表达式]

{局部变量表}中可说明 0 个或多个局部变量,说明变量时只列出变量名,不需对变量进行类型说明.局部变量名之间用逗号分隔,并可在说明时赋以初始值,当表达式为复合表达式时,表达式之间用分号相隔,并以最后一个表达式的值作为 **Module** 结构的值.

一个程序可作为一个输入行,也可保存在一个文件中,在输入行中用"<<"调入文件,就象调入系统程序包中的程序一样.在系统程序包 **Packages** 中放着一些子程序包,每个子程序包中都有若干函数,图 16-6 中列出 **Packages** 的子包 **Graphics** 的程序.

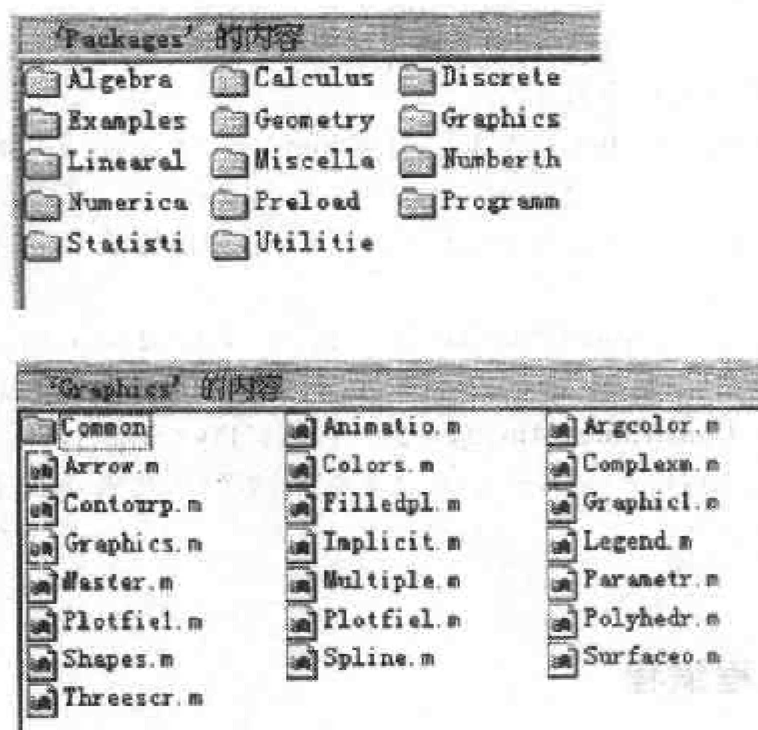


图 16－6 Packages 和 Graphics 的内容

例：调出样条插值函数对数据 p 作样条插值，并计算 s(3.7)

```
In[1]:=<<NumericaMath'SplineFit'
In[2]:=p={{0,0},{1,2},{3,-1},{1,3}{4,2}};
In[3]:=s=SplineFit[p,Cubic]
Out[3]=SplineFunction[Cubic,{0.,3.},<>]
In[4]:=s[3.7]
Out[4]={2.6515,2.8265}
```

● **条件控制结构**

逻辑表达式和逻辑运算符

关系表达式是最简单的逻辑表达式，用关系表达式表示一个判别条件．例如：$x>3$．

关系表达式的一般形式：<表达式><关系运算符><表达式>

关系运算符	==	!=	>	>=	<	<=
数学意义	=等于	≠不等于	>	≥	<	≤

逻辑表达式的一般形式：＜关系表达式＞＜逻辑运算符＞＜关系表达式＞

逻辑运算符有：!（逻辑非），&&（逻辑与、**and**），||（逻辑或、**or**）.

通常，我们已经习惯地认为逻辑表达式的值非真即假. 在 Mathematica 中，逻辑表达式的值有三个：真、假和非真非假. 当判定条件成立时，逻辑表达式的值为 True（真）；当判定条件不成立时，逻辑表达式的值为 False；当判定条件无法判断时，逻辑表达式的值非真非假. 非真非假逻辑值的是什么值？它仍然是一个逻辑表达式. 例如：

In[1]:=**x=3;x>y**

Out[2]=x>y（＊因为没有给 y 赋值，Mathematica 也就无法判断 x 是否大于 y＊）

In[3]:=**x>0**

Out[3]=True

In[4]:=**x>10**

Out[4]=False

If 语句结构

If 语句的结构与一般程序设计语言结构类似. 由于 Mathematica 的逻辑表达式的值有三个：真（True）、假（False）和"非真非假". 因此，条件语句的转向也有三种情况. 下列 **If** 结构的三种情况.

If[逻辑表达式，表达式 1，表达式 2，表达式 3]

当逻辑表达式的值是 True 时，转向计算表达式 1，当逻辑表达式的值是 False 时转向计算表达式 2，当逻辑表达式的值非 True 非 False 时（无法判定时），计算表达式 3，并将所计算表达式的值作为整个 **If** 结构的值. 其简单的形式是：

If[逻辑表达式，表达式 1，表达式 2]

If[逻辑表达式，表达式 1]

In[1]:=**g[y_]:=If[y>0,"ABC","DEF","XYZ"]**

In[2]:=**{g[7],g[-1],g[z]}**

Out[2]={ABC,DEF,XYZ}（＊z 没有赋值，逻辑表达式 z>0 的值非 True 非 False＊）

Which 语句结构

Which 语句的一般形式：

Which[条件 1，表达式 1，条件 2，表达式 2，…，条件 *n*，表达式 *n*]

Which[条件 1，表达式 1，…，条件 *n*，表达式 *n*，True，表达式]

依次计算条件 i，计算对应第一个条件为 True 的表达式的值，作为整个结构的值. 如果所有条件的值都为 False，则整个结构的值是 Null，用 True 作为 **Which** 的最后一个条件时，可用于处理其它情况，相当于 C 语言中 Switch 语句中的 default 的作用.

例如：计算

$$\begin{cases} -x, x < 0 \\ \sin(x), 0 \leqslant x < 6 \\ x/2, 16 \leqslant x < 20 \\ 0, \text{其他} \end{cases}$$

In[3]:=**h[x_]:=Which[x<0,-x,x>=0&&x<6,Sin[x],x>=16&&x<20,x/2,True,0]**

In[4]:={**h[5],h[16.2],h[z]**}

Out[4]={Sin[5],8.1,0}(* z 未赋值，不满足前三个条件 *)

In[5]:=**k[x_]:=Which[x>1,u=1,x>2,v=2,x>3,w=3]**

In[6]:=**k[6]**

Out[6]=1(* 同时满足三个条件，执行第一个条件对应的表达式 *)

Switch 语句的一般形式：

Switch[expr，模式 1，表达式 1，模式 2，表达式 2，…]

将表达式 **expr** 的值与模式 1，模式 2，…，依次做比较，给出第一个与 **expr** 匹配的模式 i 对应的表达式 i 的值. 若没有匹配的模式，则整个结构的值为 Null.

In[7]:=**g[x_]:=Switch[Mod[x,3],0,a,1,b,2,c]**

In[8]:={**g[7],g[8],g[9],g[z]**}

Out[8]={b,c,a,Null}

● 循环控制结构

Mathematica 中共有三种描述循环的语句，它们是 **Do**、**While** 和 **For**. **Do** 循环类似于 **Fortran** 中的 **Do** 语句，它根据循环描述先计算循环次数，再做循环体，常用于有确定循环次数的循环结构. 在 **While** 和 **For** 中，做一次条件确认后，做一次循环体的表达式. 类似于 C 语言中的 While 和 For 语句.

Do 语句的一般形式为：**Do**[循环体，{循环范围}]

Do 语句有下列形式：

Do[expr，{i，io，i1，s}] 循环变量 **i** 从 **i0** 到 **i1**，每次 **i** 增加 **s**，计算表达式 **expr**.

Do[expr，{i，i1}] 同上，当循环初值 **i0=1**，步长 **s=1** 时可省略不写

Do[**expr**,{*n*}]　对表达式 **expr** 计算 **n** 次

Do[**expr**,{**i**,**i0**,**i1**,**is**},{**j**,**j0**,**j1**,**js**}]　**i** 从 **i0** 到 **i1**,每次 **i** 增加 **s**,对每个 **i**,**j** 从 **j0** 到 **j1**(步长 **js**)做表达式 **expr**

In[1]:=**t=x;Do[t=1/(1+kt),{k,2,6,2}];t**

$$Out[1]=\frac{1}{1+\frac{6}{1+\frac{4}{1+2x}}}$$

In[2]:=**Do[Print[{i,"",j}],{i,3},{j,i}]**(*Out[2]略*)

While(当型)语句的一般形式:

While[条件,循环体]

当条件为 True 时则对循环体表达式求值,重复对条件判断和对循环体求值过程直到条件非真(非 True)时停止.当条件的值非 True 非 False 时,循环结构不做任何工作.使用 **While** 结构,循环体中的表达式可能一次也不做,也可能永无止境地做下去.

In[3]:=**n=19;While[(n=Floor[n/2])! =0,Print[n]]**

Out[3]=8

4

2

1

For 语句的一般形式:

For[初始值,条件,修正循环变量,循环体]

In[4]:=**For[i=1;t=x,i^2<10,i++,t=t^2+1;Print[t]]**

Out[4]=$1+x^2$

$1+(1+x^2)^2$

$1+(1+(1+x^2)^2)^2$

Mathematica 中的 **For** 和 **While** 和 C 语言中的 For 和 While 的工作方式大致相同,也许你已经发现也有不同之处,逗号和分号的作用在 Mathematica 中和在 C 语言中正好相反.

重复运用函数的方法

在符号计算语言中系统提供了一些能实现循环和迭代的函数,在计算中常用系统函数而较少用到条件语句或循环语句.一般说来,直接调用系统函数总比构造循环语句的做法简单.下列 Mathematica 的常用迭代函数及其意义.

函数	意义
Nest[f,expr,n]	对表达式 expr 作用 f 函数 n 次
NestList[f,expr,n]	从表达式 expr 运用 f 函数 0 到 n,并给出列表结果
FixedPoint[f,expr]	从表达式 expr 开始,重复运用 f 函数,直到结果不变

例如:

In[5]:=**Nest[f,x,5]**

Out[5]=f[f[f[f[f[x]]]]]

In [6] :=**FixedPoint[Function[k,Print[x];Floor[t/4]],33]**(* 最后的结果为 0 *)

退出循环结构

在 Mathematica 中退出循环结构可用下列函数:

Return[expr]退出函数中的所有过程和循环,返回值 **expr**

Break[]结束本层循环,并以 Null 为结构的值

Continue[]转向本层 **For** 或 **While** 结构中的下一次循环

Throw[expr]返回 **expr** 作为最近的外层的 **Catch** 的值

如果你用过 C 语言,你会发现 **Return**、**Break** 和 **Continue** 在 Mathematica 中的工作方式与 C 语言中的相同.

In[7]:=**f[y_,x_]:=Block[{t},t=D[y,x];If[t==0,Return["* * *"]];**

t=t^2+1;Return[t]]

In[8]:=**f[6,x]**

Out[8]=* * *

In[9]:=**t=1;Do[t *=k;Print[t];If[t>19,Break[]],{k,10}]**

1

2

6

Out[9]=24

● 程序实例

例 1:计算一组数据的算术平均值、几何平均值、中差、方差和标准偏差.

程序设计如下,并取 **Statistics.m** 文件名保存.

Mean[list_List]:=Apply[Plus,list]/Length[list](* 计算数据 list 的算术平均值 *)

GeometricMean[list_List]:=Apply[Times,list]^(1/Length[list])

(* 计算数据 list 的几何平均值 *)

```
Median[list_List]:=Module[{s1,len},
        len=Length[list];s1=Sort[list];
        If[OddQ[Length[s1]],s1[{len/2}],
        (s1[[len/2]]+s1[[len/2+1]])/2]
        ](*计算数据 list 的中差*)
Variance[list_List]:=Mean[(list-Mean[list])^2](*计算数据 list 的方差*)
'Range[list_List]:=Apply[Max,list]-Apply[Min,list](* 计算数据的标准偏差 *)
(*'Range*Hides System'Range* *)
In[1]:=<<Statistics.m
In[2]:=data=Table[Random[],{10}];
In[3]:=Mean[data]
Out[3]=0.359058
In[4]:=GemetricMean[data]
Out[4]=0.257275
In[5]:=Range[data](*Statistics' 包中的 Range 函数*)
Out[5]=0.684207
In[6]:=System'Range[6](*系统中的 Range 函数*)
Out[5]={1,2,3,4,5,6}
```

例 2:随机形成矩阵 $A_{n\times n}, n=10$ 计算

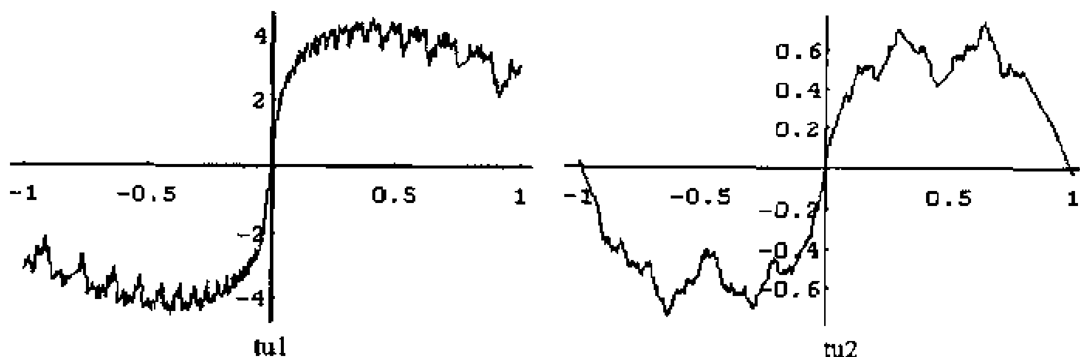
$$\| A \|_1 = \operatorname*{Max}_k \sum_{i=1}^{n} |a_{ik}|, \ \| A \|_\infty = \operatorname*{Max}_i \sum_{k=1}^{n} |a_{ik}|$$

```
In[1]:=f[M_]:=Max[Table[Apply[Plus,Abs[M[k]]],{k,1,Length[M]}]]
In[2]:=A=Table[Random[],{10},{10}](*随机形成矩阵 A10,10 *)
In[3]:=f[A](*计算 ‖A‖∞ *)
Out[3]=3.53473
In[4]:=f[Transpose[A]](*计算 ‖A‖1 *)
Out[4]=3.30627
```

例 3:作 Weierstracs 函数图象,并取不同的 r,s 观察图形.

$$f(x) = \sum_{k=1}^{\infty} r^{(s-2)k}\sin(r^k x),\ 1 < s < 2, r > 1$$

```
f[r_,s_]:=Sum[r^((s-2)k)Sin[r^k x],{k,1,100}]
ff[x_]:=f[1.2,1.5];tu1=Plot[N[ff[x]],{x,-1,1}]
ff[x_]:=f[2.2,1.2];tu2=Plot[N[ff[x]],{x,-1,1}]
```



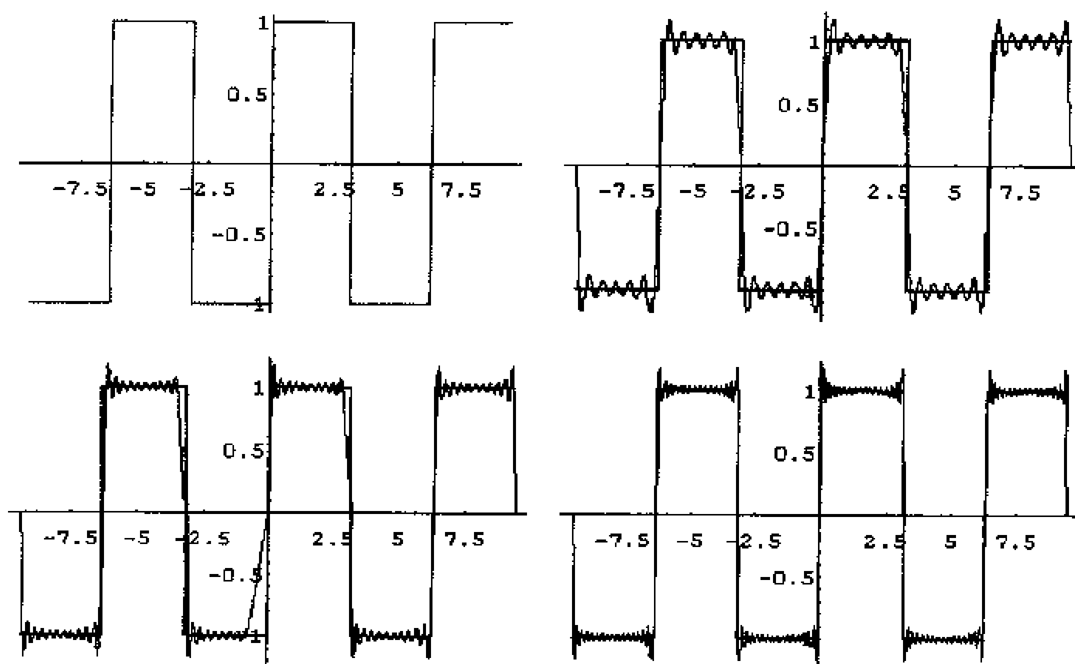
(* 图 16－7 *)

例 4：是以 2π 为周期的周期函数，

$$g(x)=\begin{cases}-1,-\pi\leqslant x<0\\1,0\leqslant x\leqslant\pi\end{cases}$$

作出的 Fourier 若干项展开和的图形，并作动画演示.

```
g[x_]:=Which[x<-2Pi,-1,-2Pi<=x<-Pi,1,-Pi<=x<0,-
1,0<=x<Pi,1,
Pi<=x<2Pi,-1,x>=2Pi,1]
t1=Plot[g[x],{x,-3Pi,3Pi},PlotStyle->{RGBColor[1,0,0]}]
For[i=1,i<=30,i=i+10;bn=(1-(-1)^n)*2/n/Pi;
h[x_]:=Sum[bn*Sin[n*x],{n,1,i}];
t2=Plot[h[x],{x,-3Pi,3Pi},DisplayFunction >Identity];
Show[t1,t2]]g[x_]:=Which[x<-2Pi,-1,-2Pi<=x<-Pi,1,-Pi
<=x<0,-1,0<=x<Pi,1,
Pi<=x<2Pi,-1,x>=2Pi,1]
t1=Plot[g[x],{x,-3Pi,3Pi},PlotStyle->{RGBColor[1,0,0]}]
For[i=1,i<=30,i=i+10;bn=(1-(-1)^n)*2/n/Pi;
h[x_]:=Sum[bn*Sin[n*x],{n,1,i}];
t2=Plot[h[x],{x,-3Pi,3Pi},DisplayFunction->Identity];
Show[t1,t2]](* 节省篇幅而将图形两个一排 *)
```



（*图16—8*）

单击图形最右边的大扩号，单击动画演示按钮 ，此时屏幕上轮流显示每一副图，产生动画的视觉效果.

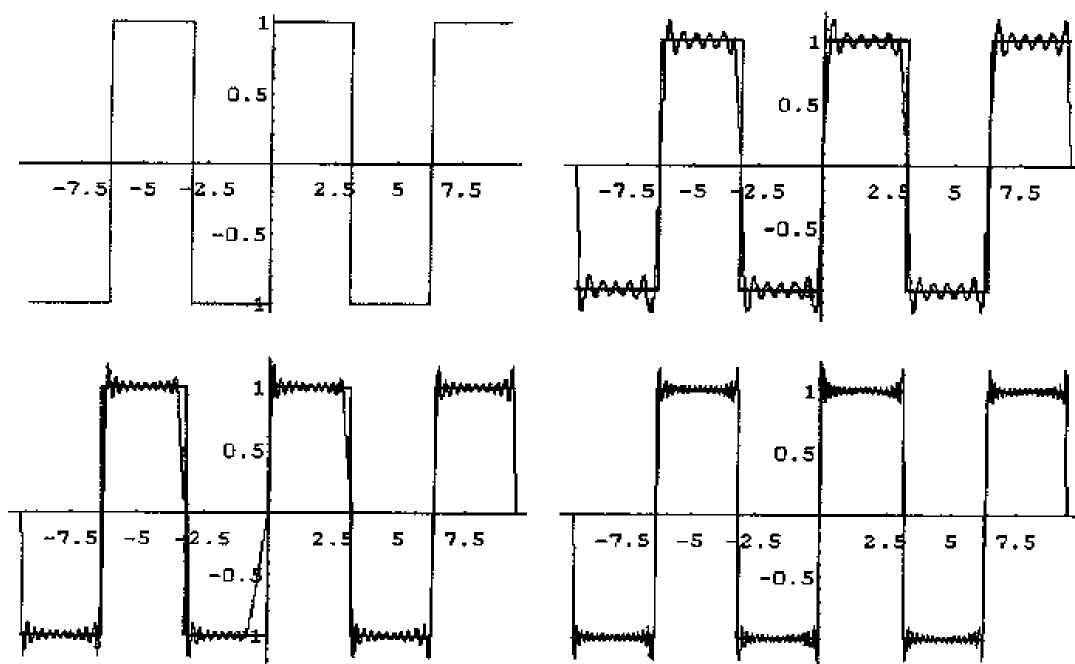
通常，单击一个图形右边的大扩号，按住Shift键单击第二个图形右边的大扩号，即选定两个图形.选定几个图形后，单击动画演示按钮 或按Ctrl+Y，则轮流显示每一副图，产生动画效果.在主菜单Options中单击Animation调出动画设置对话框，可在框上设置动画运行的方向、每秒显示的祯数等选项值.

参考书目

1. 张韵华. Mathematica 符号计算系统实用教程. 合肥：中国科学技术大学出版社，1998

2. 裘宗燕. 数学软件系统的应用及其程序设计. 北京：北京大学出版社，1994

3. 沈凤贤等. Mathematica 手册——用 IBMPC 机处理数学问题通用软件包. 北京：海洋出版社，1992



(＊图 16－8＊)

单击图形最右边的大扩号，单击动画演示按钮，此时屏幕上轮流显示每一副图，产生动画的视觉效果.

通常，单击一个图形右边的大扩号，按住 Shift 键单击第二个图形右边的大扩号，即选定两个图形. 选定几个图形后，单击动画演示按钮或按 Ctrl＋Y，则轮流显示每一副图，产生动画效果. 在主菜单 Options 中单击 Animation 调出动画设置对话框，可在框上设置动画运行的方向、每秒显示的祯数等选项值.

参考书目

1. 张韵华. Mathematica 符号计算系统实用教程. 合肥：中国科学技术大学出版社，1998

2. 裘宗燕. 数学软件系统的应用及其程序设计. 北京：北京大学出版社，1994

3. 沈凤贤等. Mathematica 手册——用 IBMPC 机处理数学问题通用软件包. 北京：海洋出版社，1992